ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERÍA DE TELECOMUNICACIÓN

Título del proyecto:

"Técnicas de Seguimiento de Puntos Faciales y su Efecto en la Estimación de la Posición 3D de la Cabeza"

David Roncal Redín

Tutores: Rafael Cabeza Laguna

Mikel Ariz Galilea

Pamplona, 29 Abril 2014

Agradecimientos

Aunque haya conseguido escribir el resto del proyecto más fácilmente, esta página es posiblemente, la que más me ha costado realizar. Hay tantas personas que han formado parte de mí y a las que tengo que agradecer que pueda estar hoy aquí, que no sé ni si podré resumir mi agradecimiento en unas frases.

Primero, y empezando por el final, tengo que agradecer a mis tutores, Rafa y Mikel, por su tranquilidad, su constancia y su buen quehacer, siempre con una disposición rápida y tremendamente positiva. Gracias a ellos este proyecto ha salido adelante, y son merecedores de más de la mitad del mérito. Asimismo agradecer a José Javier, a Jorge y a Cristian, y a todos los que han formado parte de este laboratorio conmigo por su calidad personal.

Segundo, no puedo dejar de mencionar a mi familia, la cual ha formado parte desde el principio de mi vida en mi formación, en mi educación en la persona que soy hoy en día. Es difícil plasmar lo importante que es para mí la suerte de tener a mis padres y a mis hermanos siempre apoyándome y animándome a seguir. Tener un colchón así es algo imposible de valorar o incluso de devolver.

También tengo que dar las gracias a Andrea, por creer en mí, sin importar lo que pasara. Por destacar mis cualidades y por saber confiar en la persona que soy, rompiéndome mis esquemas y enseñándome que hay vida más allá de mi mundo, y que es nuestro deber intentar hacer algo por los demás.

Aunque destaco a esta persona tan especial, forma parte de un grupo aún más grande, entre los que están mi familia Scout y mi Cuadrilla, ambos muy heterogéneos pero muy importantes también. Los Scouts por devolverme la pasión por la vida, por las sonrisas con los chavales y el hacerme creer que puedo dejar mi semilla en alguien. Mi Cuadrilla por valorarme, por entenderme y por quererme. Por saber sacar el lado positivo de lo que hago y pincharme, sabiendo siempre que están ahí.

Aun con todo, me dejo muchísimas personas que han intercambiado conmigo sus ideas, que me han enseñado valores: mi colegio, mis profesores, mis amigos en USA, mis compañeros de carrera, mis varios equipos de futbol…

A todos vosotros os doy gracias, en cierta medida todos habéis colaborado en este proyecto.

1. Introducción

En los últimos años las novedades en el ámbito de las interfaces hombre-ordenador han sido cuantiosas al introducirse nuevas maneras de interaccionar. Concretamente la disponibilidad de datos sobre la posición del usuario e incluso de sus gestos faciales ha abierto nuevas vías de comunicación. Así, a través de la webcam y el seguimiento de la mirada se reconocen diferentes tipos de expresiones faciales, y se detecta a qué punto de la pantalla se está enfocando la vista. El proyecto que se propone se enmarca en esta línea de trabajo.

Concretamente se estudiarán diferentes algoritmos de seguimiento de puntos faciales para su utilización posterior en sistemas de detección de la posición de la cabeza. Los algoritmos a estudiar estarán basados en su mayoría en el estándar propuesto por Lucas-Kanade[3]. Con el objeto de estudiar estos algoritmos se usará una base de datos de vídeos anotada en cada uno de los fotogramas para comprobar el error de seguimiento de los algoritmos bajo estudio.

Para esta base de datos se utilizará un maniquí dotado de rasgos faciales realistas y un número determinado de usuarios reales. En el busto se realizará un escaneo tridimensional teniendo así su modelo exacto. Estos vídeos y el modelo numérico se fusionarán con los datos obtenidos de un sensor magnético 6DOF para poder etiquetar cada fotograma de manera precisa.

1.1. Algoritmos de seguimiento

En este proyecto son de vital importancia (son la base del proyecto) los algoritmos de seguimiento. Estos no son más que diferentes formas de realizar computacionalmente el seguimiento de un objeto en un vídeo, es decir, la trayectoria que éste realiza a lo largo del tiempo en la imagen que se está grabando de él.

Existen multitud de algoritmos de seguimiento o *tracking*, pero los que se van a tratar aquí son: Lucas-Kanade, algoritmos basados en descriptores, AAM, ASM e IntraFace.

1.1.1. Lucas-Kanade

El algoritmo de Lucas-Kanade (LK) surgió ante la necesidad de reducir el coste computacional que tenían otros algoritmos de seguimiento. Esta reducción se produce gracias a una serie de técnicas que buscan un alineamiento utilizando menos puntos

potenciales que otros algoritmos. Por ejemplo, para el Registro Exhaustivo se requiere $O(M^2N^2)$ tiempo de computación, mientras que utilizando Lucas-Kanade se requiere $O(M^2 \log N)$, lo cual mejora el tiempo consumido. El algoritmo fue creado por Bruce D. Lucas y Takeo Kanade[3] en 1981.

Para entender mejor el algoritmo, se explicará cómo funciona primero en una dimensión. El objetivo es ver cómo varía una función f(x) con respecto a otra g(x) que se suponen prácticamente iguales en forma pero desplazadas en el espacio:

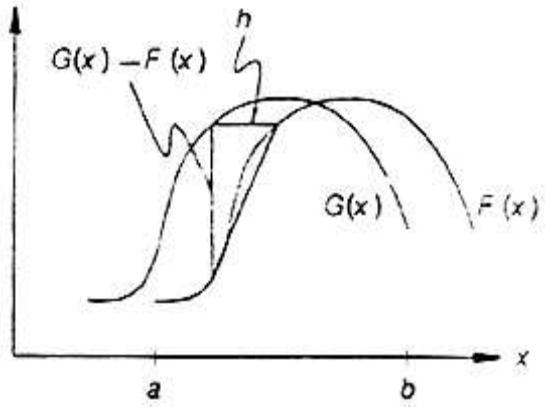


Figura 1: Acercamiento de una función a otra con Lucas-Kanade

Así, se trata de averiguar el valor h que da el mejor parecido entre G y F. La solución a este problema depende de una aproximación lineal del comportamiento de f(x) alrededor de valores de x. Así:

$$F'(x) = \frac{F(x+h) - F(x)}{h} = \frac{G(x) - F(x)}{h}$$ y así se deduce:

$$h \approx \frac{G(x) - F(x)}{F'(x)}$$

El éxito de este algoritmo depende de que h sea lo suficientemente pequeño como para que la aproximación sea adecuada. A partir de un valor de h que se da aleatorio pero pequeño, se estima el valor más cercano y así, utilizando un método iterativo se combinan todas las h para calcular la más cercana al mejor registro.

El problema de éste método es que no se generaliza fácilmente para dos dimensiones, ya que cada una de las aproximaciones en cada dimensión ocurre de diferente forma. Para solucionar esto, se aplica la aproximación lineal a esta ecuación:

$$F(x+h) = F(x) + h \cdot F'(x)$$

Para encontrar el valor de h que minimiza la medida normal de la diferencia entre las curvas

$$E = \sum_x [F(x+h) - G(x)]^2$$

Se deriva E y si obtiene el valor que lo convierte en cero para que sea un mínimo, de tal manera que

$$h \approx \frac{\sum_x F(x)[G(x) - F(x)]}{\sum_x F(x)^2}$$

A partir de esto y realizando una convergencia entre varias iteraciones, se encuentra el valor de h más pequeño tomando una serie de pesos entre los puntos investigados. Este método se puede generalizar para más dimensiones.

Por último es necesario añadir que la implementación de este algoritmo se hace, habitualmente, de manera piramidal:

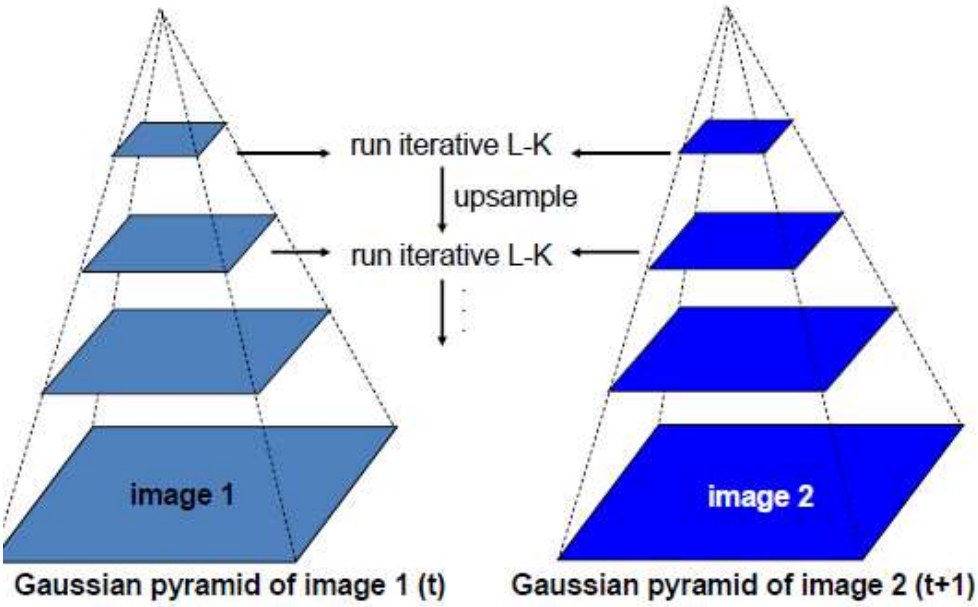


Figura 2: Pirámides de búsqueda de Lucas-Kanade

Esto significa que se analiza el flujo óptico a distintas resoluciones. Primero se comienza con el nivel más profundo de la pirámide, que corresponde a la resolución más pequeña, de manera que es capaz de localizar flujos más grandes. Una vez hecha la estimación a ese nivel, se utiliza como inicialización del nivel anterior, y así sucesivamente hasta llegar a la resolución más grande para afinar al máximo detalle.

1.1.2. Descriptores

El primer paso es definir qué es un descriptor. Un descriptor es un vector característico de la imagen que se está tratando y que tiene una serie de datos incluidos en él. El primero es el de la localización, pero después *describe* la región en la que está, de tal manera que se fija en una serie de puntos de la vecindad y contiene una serie de valores a partir de ellos.

En resumen, cada punto tiene asociado un descriptor del tipo que se plantee, ya que a partir de su vecindad se pueden obtener una serie de datos tales como la desviación estándar, la media, etc. de la ventana en la que está. Sin embargo, cuanto más característica es la región en la que se encuentra (las esquinas de los ojos, las comisuras de los labios) más característico es el punto y se convierte en un descriptor que destaca entre los demás.

Para la detección de puntos característicos y creación de descriptores asociados a ellos existen muchos algoritmos implementados que calculan diferentes datos sobre la región. En este proyecto se van a destacar varios de ellos: el método BLOCK (por defecto en Matlab), el SURF, el SIFT y el FREAK.

1.1.2.1. BLOCK

El método BLOCK viene implementado por Matlab. Una vez se tiene la imagen y los puntos de cuáles hay que extraer las características que formaran el descriptor, lo primero es comprobar que están dentro de los límites marcados como tamaño de bloque.

Si están dentro del tamaño de bloque que se está calculando, se obtienen los valores del cuadrado que rodea el punto y esto será su descriptor.

1.1.2.2. SIFT

El método SIFT[13] (*Scale Invariant Feature Transform*) transforma los datos de cada imagen en coordenadas invariantes relativas a las características locales en cada punto. De esta manera se generan gran número de características que cubren la imagen sobre todo el rango de escalas y localizaciones. Por ejemplo, una imagen 500x500 píxeles puede llegar a tener 2000 puntos característicos, lo cual es importante para el reconocimiento de objetos.

El SIFT se divide particularmente en cuatro pasos bien marcados. El primero es detectar qué puntos tienen las escalas más marcadas, para lo cual se busca entre todas las escalas y localizaciones de la imagen utilizando una función de diferencia de Gaussianas, de tal manera que se hace eficientemente. Una vez se han hallado los puntos de mayor interés y que son invariantes en escala y orientación, se determinan las localizaciones de dichos puntos.

El tercer paso es asignar la orientación u orientaciones que tiene cada punto destacado. Se obtienen utilizando las direcciones de los gradientes locales alrededor de cada punto. Esta orientación es crucial, ya que las futuras operaciones se hacen basándose en ella.

Por último y obtenidas la localización, la escala y los gradientes de la orientación, se crean los descriptores a partir de ellos, con la escala de la región que utiliza cada punto. Así se transforman los gradientes en una representación que permite dar un nivel significativo de la forma local, la distorsión y los cambios en iluminación alrededor del punto.

Se puede observar el ejemplo que utiliza el creador del SIFT, David G. Lowe, en su artículo sobre él. La primera imagen es en la que se va a aplicar el SIFT, después se obtienen los primeros puntos con sus orientaciones y localizaciones en b), por último se pasa un umbral en c) y otro en d) para obtener los puntos más significativos.

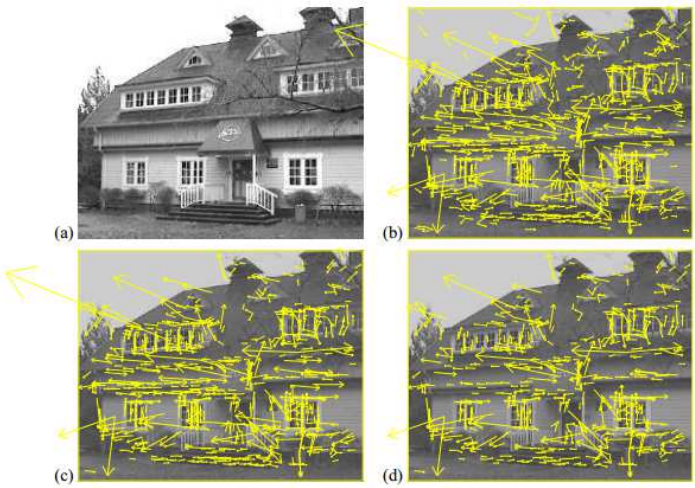


Figura 4: Procedimiento de trabajo del algoritmo SIFT

1.1.2.3. SURF

Otro de los métodos con el que se van a calcular descriptores es el SURF[11], (*Speeded Up Robust Features*). Esta técnica fue presentada por Herbert Bay en 2006 y es muy utilizada en procesado de imagen y reconstrucción 3D. Está inspirada parcialmente en el método SIFT pero es varias veces más rápida y más robusta con respecto a transformaciones de imágenes. Está basada en sumas de 2D Haar *wavelets*.

El primer paso de este descriptor consiste en conseguir una orientación basada en información adquirida a través de una sección circular alrededor del punto de interés. Después se construye una región cuadrada alineada con la orientación obtenida en el paso anterior y se consiguen las características del descriptor desde esta última región. Se va a explicar brevemente ambos procesos, el de orientación y el de obtención de características.

En el proceso de orientación, es importante que sea invariante con respecto a las rotaciones. Para ello se calculan las respuestas de las Haar-Wavelet en las direcciones x e y en un vecindario circular. Una vez se han calculado las *Wavelets* y se han ponderado adecuadamente, se representan y se calcula la orientación predominante.

Después, se construye la región cuadrada centrada alrededor del punto de interés y orientada según el vector obtenido con anterioridad. Esta región se divide hasta conseguir 4x4 regiones cuadradas y se hacen una serie de cálculos con las respuestas de las *Wavelets* tales como los valores medianos y medios, etc. En la siguiente figura se pueden observar los pasos dados, a la izquierda las regiones circulares para averiguar la orientación y a la derecha las regiones cuadradas desde las que se obtienen los datos que van a definir el descriptor.

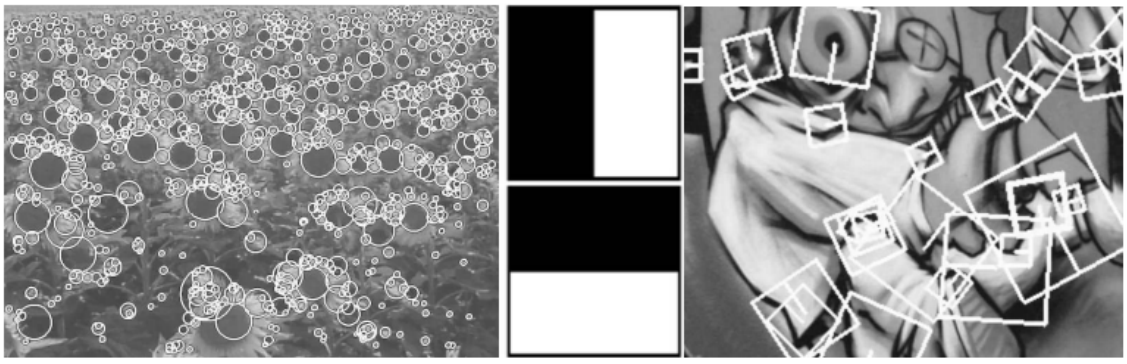


Figura 3: Procedimiento de trabajo del algoritmo SURF

1.1.2.4. FREAK

El método FREAK[12] (*Fast REtinA Keypoint*) para el cálculo de descriptores también es muy utilizado en sistemas de seguimiento. Está fundado en el funcionamiento de la retina del ojo humano cuando compara intensidades de colores.

La retina de los seres humanos no observa el mundo en una posición quieta y calmada, si no que el ojo se mueve con movimientos discontinuos. La tipología presentada en el FREAK intenta simular dichos movimientos, pero además la retina en un primer barrido rápido desecha gran parte de la información que no se parece a lo que está buscando, de tal manera que enseguida va centrándose en los pocos elementos que pueden ser parecidos a las características o descriptores que busca en función de las intensidades.

Para comparar intensidades hay muchos tipos de rejillas que se pueden tomar. Por ejemplo BRIEF y ORB, otro tipo de métodos, toman puntos por parejas de manera aleatoria. BRISK usa un modelo circular dónde los puntos están igualmente espaciados en círculos concéntricos. Para el uso del FREAK lo que se implementa es un muestreo circular también, pero otorgando mayor densidad de puntos cerca del centro. Esta densidad disminuye de manera exponencial y radial.

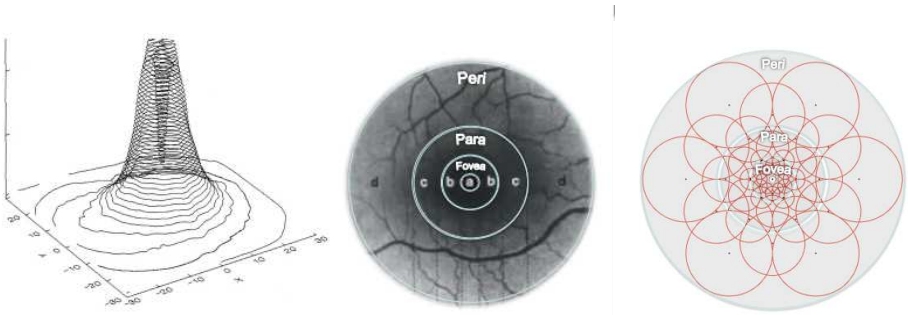


Figura 5: Procedimiento de trabajo del algoritmo FREAK

Una vez se obtienen los puntos que se van a describir se van tomando diferentes tamaños de región alrededor de los puntos, más grandes cuanto más lejos están del centro. También se utiliza una superposición entre ellos para mejorar la actuación y lograr los descriptores más característicos. A partir de estas regiones, se calculan una serie de datos usando parejas de campos y se crea una matriz con los puntos más significativos en cuanto a media y varianza. Cuanta más alta es la varianza mejor es el descriptor.

Por último, el FREAK también implementa una orientación a partir de cálculos de gradientes locales alrededor de las imágenes.

1.1.3. IntraFace

IntraFace[15] es un programa desarrollado por la Universidad de Pittsburgh y el Carnegie Mellon. Fue apoyado por la Fundación Nacional de Ciencia Norteamericana y por la Oficina de Laboratorios de Desarrollo Naval.

Se trata de un algoritmo disponible públicamente en C++ o en Matlab que detecta atributos faciales, detecta puntos característicos en la cara (en las cejas, ojos, nariz y boca), también analiza expresiones faciales y puede transferirlas a otras imágenes.

En este proyecto se utilizará el IntraFace para obtener los puntos de las bases de datos de vídeos y estos puntos se introducirán con el resto del programa para averiguar la posición y rotación de la cabeza.

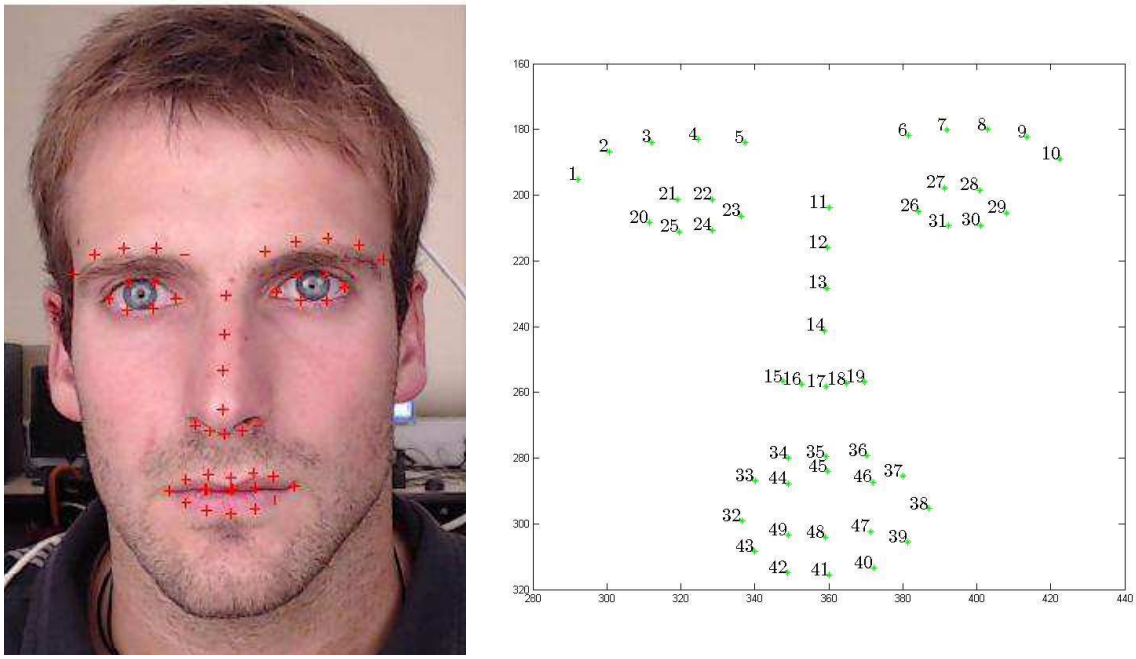


Figura 6: Puntos del IntraFace y su relación de índices

En la imagen de la izquierda se observan los puntos del IntraFace en rojo. En la imagen de la derecha se ve el orden de los índices del IntraFace.

1.1.4. *Active Shape Models* (ASM)

ASM es un método que trabaja con diferentes modelos estadísticos sobre la forma del objeto que se está estudiando. Se trata de deformarla iterativamente para adecuarla al

tamaño del objeto en la nueva imagen. Fue desarrollado por Tim Cootes y Chris Taylor en 1995[14].

Las imágenes varían sólo de la manera en la que se han entrenado al conjunto de modelos estadísticos, y la forma del objeto se representa con un grupo de puntos controlado por el modelo entrenado. En este caso, se entrenan las imágenes basándose en órganos faciales.

De manera básica, ASM funciona alternando dos pasos básicos:

- Genera una forma inicial observando en la imagen los alrededores de cada punto y buscando la mejor posición.
- Una vez se tiene la forma inicial, se modela con la distribución de puntos del modelo.

En la imagen inferior se puede observar estos dos pasos:

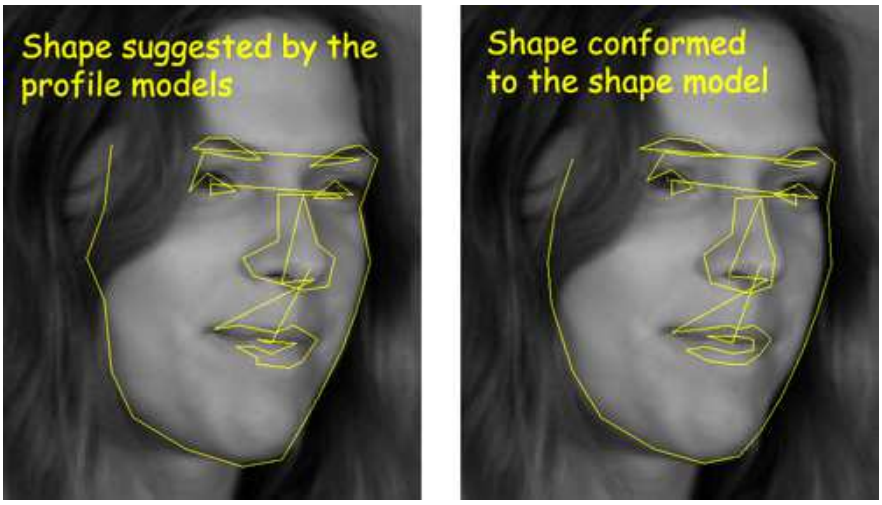


Figura 7: Imágenes de funcionamiento de ASM

En la implementación que se utiliza en este proyecto, realizada por José Javier Bengoechea[19], se trabaja de la siguiente forma.

Para la construcción del modelo de apariencia se utilizan los puntos marcados en la base de datos y que se consideran característicos. Alrededor de ellos se toma un perfil de apariencia de manera perpendicular al contorno del objeto. Una vez se obtienen todos los perfiles se aplica PCA para caracterizar esa apariencia de la manera más eficiente posible.

Una vez obtenido el modelo, para cada imagen nueva se realiza una segmentación. Ésta consiste en localizar aquellos puntos de la imagen cuya forma es coherente con el

modelo estadístico. Esta medida se cuantifica con la distancia de Mahalanobis entre la nueva imagen y el modelo. El algoritmo está detalladamente explicado en la memoria de José Javier.

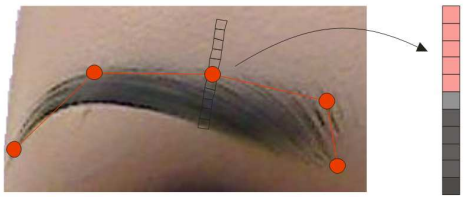


Figura 8: Región para el modelo ASM

1.1.5. *Active Appearance Model* (AAM)

AAM es un algoritmo de Seguimiento que une la forma y la apariencia de un objeto a la nueva imagen. Estos modelos se crean en una fase anterior de entrenamiento a partir de un grupo de imágenes con las marcas que se van a seguir en cada una.

El algoritmo fue presentado por Edwards, Cootes y Taylor en la 3ª Conferencia Internacional de Reconocimiento de Cara y Gesto en 1998[17]. Este algoritmo se utiliza ampliamente en temas de interpretación de imágenes médicas.

Este algoritmo usa la diferencia entre la actual estimación de la apariencia y la imagen objetivo y comienza un proceso de optimización. Utiliza técnicas de errores cuadráticos mínimos y calcula las nuevas imágenes a gran velocidad. Este tipo de modelo está muy relacionado con el ASM, pero éste método tiene la ventaja de que puede aprovechar toda la información disponible de la textura a través del objeto modelado.

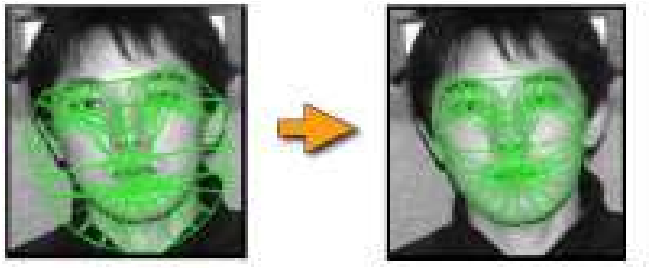


Figura 9: Imágenes de funcionamiento de AAM

En este caso también se implementa cómo en ASM, pero las regiones de entrenamiento no son perpendiculares a los puntos característicos, sino que AAM triangula la imagen entre ellos y contempla la apariencia de la región contenida en ellas. La manera de proceder de este algoritmo en cuanto al cálculo de las regiones se observa en la **figura 10.**

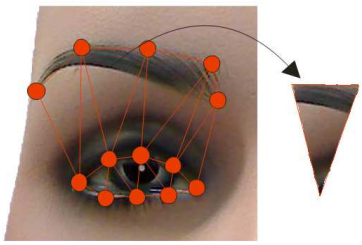


Figura 10: Regiónes en AAM

1.2. POSIT

POSIT[7] es un método para buscar la pose de un objeto utilizando una imagen. Para que se pueda implementar, se asume que se pueden detectar cuatro o más puntos en la imagen y que se conocen sus coordenadas relativas en el modelo geométrico de 3D. Es decir, se tienen como mínimo cuatro puntos en la imagen y sus correspondientes puntos en tres dimensiones del objeto. También se presupone que estos puntos no son coplanarios, esto es, no forman un plano.

Este método combina dos algoritmos, primero el POS (*Pose from Orthography and Scaling*), el cual aproxima la proyección perspectiva con una proyección escalar del objeto y encuentra la rotación y la translación del objeto resolviendo un sistema lineal. El segundo algoritmo es el POSIT propiamente dicho, que es "POS with ITerations". Esto implica que se utilizan varias iteraciones para aproximar la pose y obtener la mejor proyección ortográfica.

El punto a favor de este algoritmo es que no necesita empezar con una suposición inicial, y calcula la pose con un orden de magnitud más bajo que otros algoritmos. En caso de utilización en tiempo real puede ser muy útil puesto que el tiempo de computación es muy reducido. Además, el código está escrito en pocas líneas.

El cálculo de la posición y orientación de un objeto usando imágenes tiene aplicaciones muy importantes, tales como calibración, cartografía, seguimiento y reconocimiento de objetos entre otras.

Este algoritmo forma parte de este proyecto de manera muy importante. En la evaluación de la estimación de la pose de la cabeza o HPE (*Head Pose Estimation*) utilizaremos básicamente este algoritmo en Matlab a partir de los puntos que obtengamos en el seguimiento de los objetos a lo largo de las imágenes.

1.3. Aplicaciones

En general, el seguimiento de objetos en una imagen tiene numerosas y muy diferentes aplicaciones[8]. Algunas de las más importantes son:

- **Medios de comunicación y realidad aumentada.** El seguimiento de objetos es un elemento importante en la post-producción y captura de movimiento para las industrias del cine y la televisión.

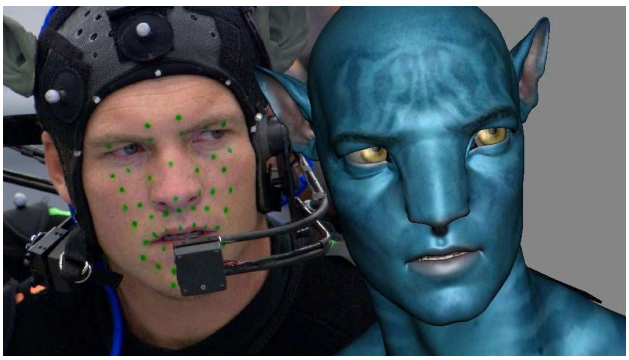


Figura 11: seguimiento de la cara para la película

- **Aplicaciones médicas e investigación biológica.** En general, el seguimiento de objetos ha sido cada vez más utilizado por sistemas médicos para ayuda en el diagnóstico y acelerar la tarea del cirujano. El seguimiento de objetos puede estimar la posición de determinados tejidos blandos o de instrumentos como por ejemplo agujas durante al cirugía.

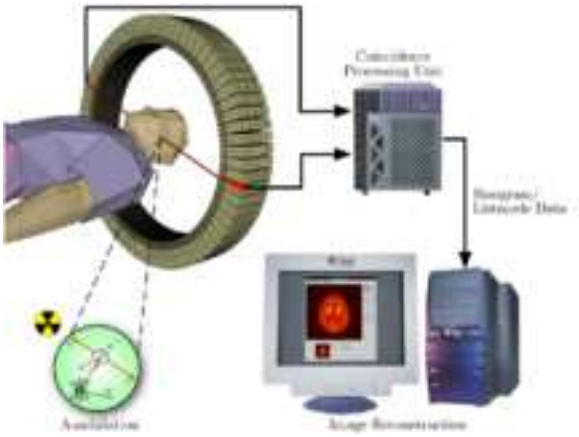


Figura 12: Funcionamiento de PET, Tomografía por Emisión de POSITrones

- **Vigilancia e inteligencia de negocios.** El seguimiento de objetos es una herramienta adecuada, utilizada en la vigilancia automática de vídeo para la seguridad, la vida asistida y las aplicaciones de inteligencia de negocio.

- **Tele-colaboración y juegos interactivos.** Las cámaras web estándar ya incluyen el software de seguimiento que localiza y sigue la cara de un usuario para videoconferencias desde el escritorio. Por otro lado, el seguimiento de ojos se utiliza para estimar el contacto visual entre los asistentes de una reunión y así mejorar la eficacia de la interacción en videoconferencias. El seguimiento de objetos también está cambiando la manera de enviar el control a las máquinas. Esta modalidad de interacción natural se utiliza en juegos interactivos.

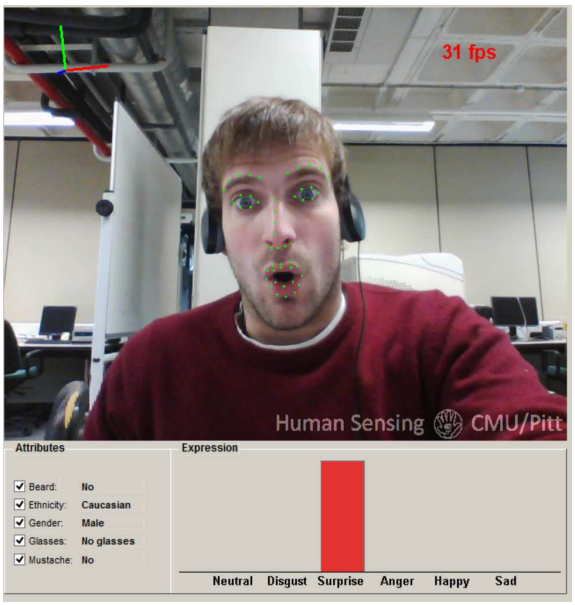


Figura 13: IntraFace sobre un usuario

- **Instalaciones de arte y espectáculos.** El seguimiento de objetos se utiliza cada vez más en instalaciones de arte y en actuaciones donde la interacción es posible gracias al uso de cámaras de vídeo y a menudo por los sistemas de proyección. La interactividad puede ser utilizada para mejorar la narrativa de una obra o para crear acciones inesperadas o reacciones del entorno.

De todas estas aplicaciones, la que más interesa es aquella en la que se cita el seguimiento de ojos en relación a la cámara web de los ordenadores, de tal manera que en todo momento se sepa dónde está mirando el usuario, tanto al ver un anuncio como en el propio sistema operativo. A partir de aquí el estudio nos puede servir para obtener gran cantidad de datos, desde las partes en las que más se fija el cliente en un anuncio, en un videoclip, hasta analizar la prioridad de dónde y en qué orden está mirando.

2. Antecedentes y Objetivos.

La capacidad[2] de estimar la posición de la cabeza de una persona es una habilidad común al ser humano que representa un reto único para los sistemas de visión por ordenador. Los seres humanos, desde tiempos inmemoriales, interpretan sin gran esfuerzo el movimiento y la orientación de las cabezas de otros seres, permitiéndoles inferir las intenciones de los sujetos y comprendiendo y desarrollando una forma de comunicación no verbal.

En el contexto de visión por ordenador, la estimación de la posición de la cabeza (*Head Pose Estimation*) es el proceso de inferir la orientación de una cabeza humana a partir de imágenes digitales. Requiere de una serie de pasos y técnicas que transforman la representación de píxeles que posee una imagen en un concepto de dirección. Como en otros procesos de visión por ordenador, un estimador ideal debe ser correcto sin importar el conjunto de cambios que puede tomar una imagen, tales como la distorsión de la cámara, la proyección y la geometría, la iluminación, las expresiones faciales o la presencia de accesorios como gafas, sombreros, pelo largo, etc.

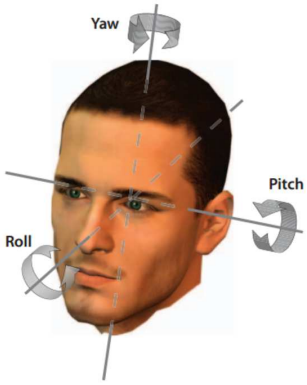


Figura 14[2]: Rotaciones de la cabeza

En el caso del ser humano, se asume que la cabeza está limitada a tres grados de libertad en cuanto a la orientación, que se definen como *Pitch*, *Roll* y *Yaw*. Además, la estimación de la posición de la cabeza está muy ligada con la estimación de la dirección de la mirada. Se ha demostrado que ésta depende de la posición de la cabeza y la dirección de los ojos conjuntamente.

Approach
Appearance Template Methods
• Image Comparison
• Filtered Image Comparison
Detector Arrays
• Machine Learning
• Neural Networks
Nonlinear Regression Methods
• Regression Tools
• Neural Networks
Manifold Embedding Methods
• Linear Subspaces
• Kernelized Subspaces
• Nonlinear Subspaces
Flexible Models
• Feature Descriptors
• Active Appearance Models
Geometric Methods
• Facial Features
Tracking Methods
• Feature Tracking
• Model Tracking
• Affine Transformation
• Appearance-based Particle Filters
Hybrid Methods
• Geometric & Tracking
• Appearance Templ. & Tracking
• Nonlinear Regression & Tracking
• Manifold Embedding & Tracking
• Other Combinations

Figura 15[2]: Métodos para la detección de la posición de la cabeza.

Gracias a estas estimaciones, podemos decir que se pueden inferir muchos comportamientos humanos. Por ejemplo, movimientos rápidos de la cabeza podrían equivaler a signos de sorpresa o alarma. También se puede observar dónde está el foco de atención de cada persona según hacia donde está enfocada su cabeza. Incluso puede proveer de información sobre el ambiente, si una persona gira su cabeza en una dirección específica, hay grandes probabilidades de que allí haya un objeto de gran interés. En conclusión, la estimación de la posición de la

cabeza podría rellenar huecos existentes entre la interacción de las personas y los ordenadores.

Se han realizado numerosos estudios sobre la estimación de la posición de la cabeza, entre los cuales hay también cuantiosos métodos para calcularla. Algunos de ellos se pueden observar en la **figura 15**.

En cuanto al objetivo de este proyecto, se pueden destacar dos bloques. El primero y bastante importante es crear un modelo en 3D del sujeto "Dasha", una cabeza de maniquí que se utiliza en el laboratorio para varios proyectos. Una vez realizado esto, también es de gran interés el cálculo de los puntos característicos del modelo, de tal manera que sepamos en qué posición se encuentran puntos como las comisuras de los labios, los córneres de los ojos o los agujeros de la nariz, entre otros en el propio modelo digital.

El segundo bloque es la evaluación de algoritmos de seguimiento para la obtención de la estimación de la cabeza. Esta sección es la más extensa, ya que trata de implementar diferentes algoritmos, en general Lucas-Kanade con diferentes variantes, tales como descriptores, AAM, ASM, y la evaluación de un algoritmo comercial como el IntraFace. Con este segundo apartado, se va a realizar un estudio y averiguar qué algoritmos realizan un seguimiento más preciso y cuál es su efecto al estimar la posición de la cabeza, objetivo del que trata todo el proyecto.

Así, a través de los datos obtenidos se calculará qué algoritmo es más eficiente, más preciso y robusto.

3. Desarrollo del trabajo.

Como se ha explicado anteriormente, el desarrollo del trabajo se puede dividir en dos grandes módulos, el escaneado y la obtención de puntos del modelo 3D y la implementación y evaluación de algoritmos. No obstante, se explicarán de manera temporal a su realización, primero se concibió el escaneado y el registro, después los algoritmos y sus correspondientes resultados.

3.1. Equipamiento

3.1.1. 3D Guidance TrakSTAR de Ascension Technology Corporation

El sistema TrakSTAR se utiliza para efectuar el seguimiento magnético y está compuesto por un transmisor y varios sensores. A partir del transmisor permite conocer la posición y orientación de cada sensor con respecto al sistema de coordenadas del mismo.

Como nexo de unión entre el transmisor y el receptor existe una unidad electrónica que se encarga del procesado de la información, controla la señal del transmisor y lo alimenta con corriente eléctrica.

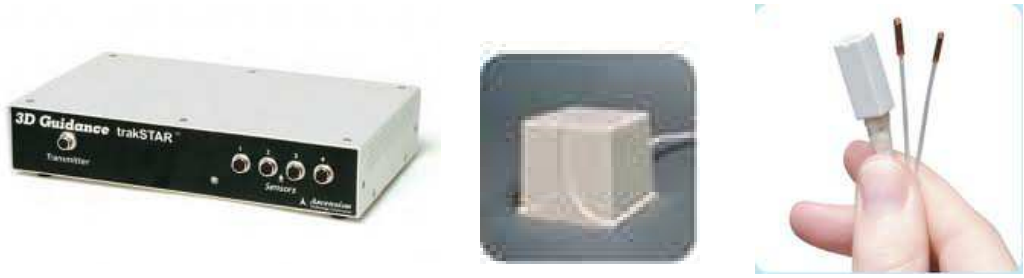


Figura 16: Conjunto de elementos de trakSTAR . A la izda. la unidad electrónica, en el centro el transmisor magnético y a la derecha los receptores.

Figura 17: Máquina Konica Minolta

3.1.2. Konica Minolta

La Konica Minolta es una máquina que se utiliza para escáneres en 3D. Es un dispositivo bastante pequeño y avanzado, y permite escanear objetos de pequeño y gran tamaño, con alta precisión y exactitud. Se monta sobre un trípode y se conecta con un ordenador, de tal manera que posee una gran variedad de aplicaciones.

Tiene una precisión de 50 micras, captura detalles del color, posee varias lentes que se pueden cambiar entre sí y es un accesorio bastante manejable. Colocado sobre un trípode te ofrece bastante variabilidad y funcionabilidad.

3.1.3. David LaserScanner

David LaserScanner es una tecnología que permite escanear objetos con un láser que se puede mover de manera manual. Presenta una manera muy fácil y cómoda de escanear objetos 3D y la relación precio – calidad es muy buena.

El kit viene con un láser que tiene foco ajustable, varios colores (verde y rojo), paneles de diferentes tamaños para realizar la calibración y el software necesario para utilizarlo en un ordenador.

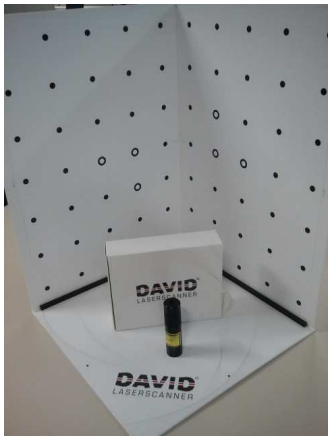


Figura 18: Aparatos DAVID LaserScanner

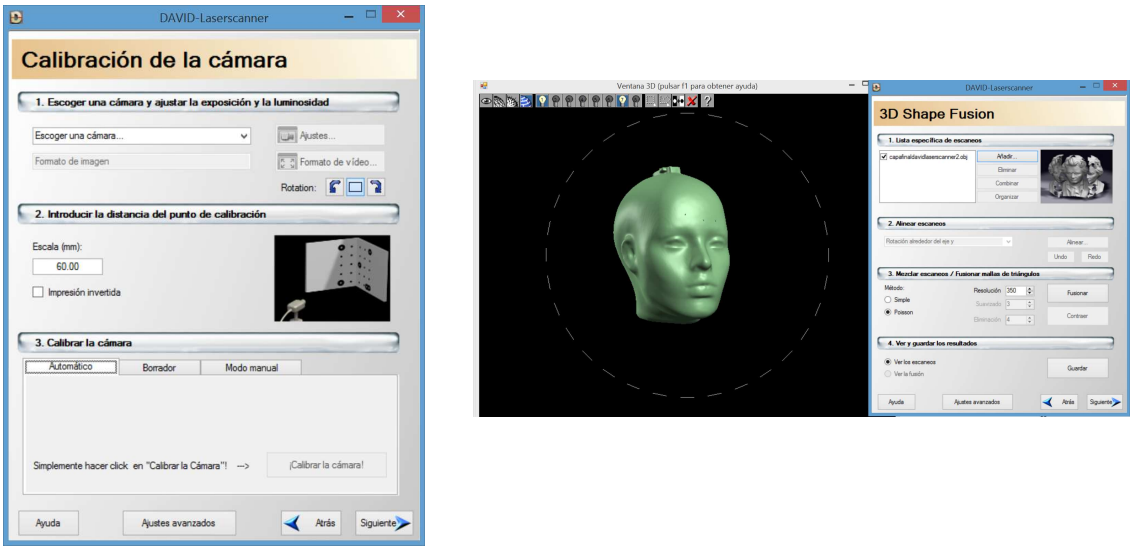


Figura 19: Ventanas del Software de DAVID LaserScanner para manejar la calibración de la cámara y luego el ensamblaje de los *patches*.

Con este elemento se ha hecho uno de los escaneos del sujeto “Dasha” en 3D.

3.1.4. Cámara Web Logitech HD Pro C920

Para la realización del proyecto ha sido necesaria tanto en la grabación de los vídeos como en varias pruebas (Lucas-Kanade, IntraFace, etc) una cámara web.

La utilizada ha sido el modelo Logitech HD Pro C920, que tiene una resolución máxima (sin interpolación) de 1920x1080 para imágenes estáticas y de 1280x720 para vídeo, a una frecuencia de hasta 30 fotogramas por segundo. Este tipo de cámaras se utilizan para filmar vídeos e imágenes de alta calidad.

Figura 20: Cámara Web Logitech HD Pro C920

3.1.5. Cabeza artificial

Una de las partes del proyecto consistía en grabar varias secuencias de vídeo con este busto y también realizarle un escaneo 3D.

Es una cabeza de un maniquí cortada y amoldada para su uso, con una pieza para sujetar el sensor en su parte más alta realizada con una impresora 3D y con un panel de plástico en su base para sujetarla a un trípode.

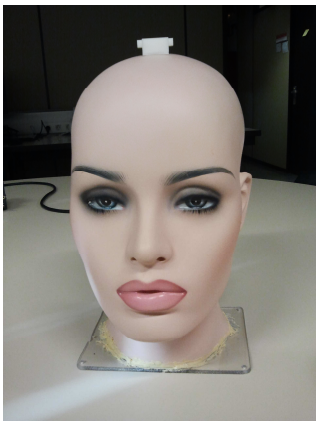


Figura 21: Modelo Dasha

3.1.6. Ordenadores

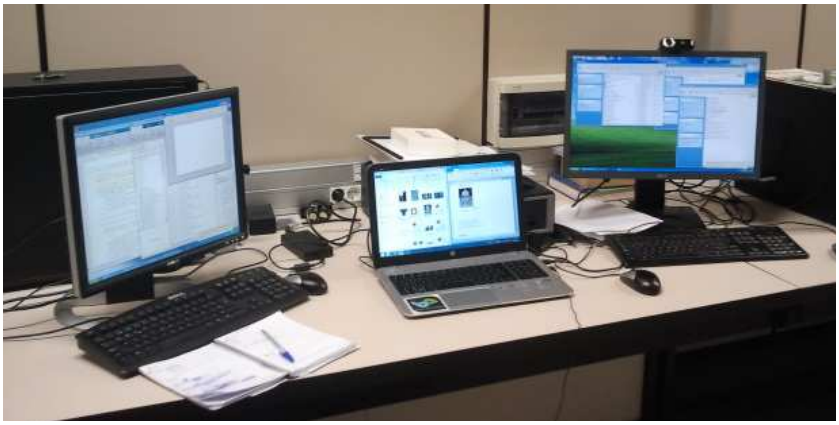


Figura 22: Ordenadores utilizados durante una simulación

Para la consecución del proyecto se han utilizado varios ordenadores con características diferentes en las simulaciones. Se comentará superficialmente sus características:

- Hp n3510
  - Intel Core i7 – 4700MQ @ 2.4 GHz
  - RAM: 8 GB
  - 64 bits

- DELL
  - Intel Core 2 Quad CPU Q8300 @ 2.5 GHz
  - RAM: 3GB
  - 32 bits

- DELL
  - Intel Xeon CPU W5590 @3,33GHz
  - RAM: 12GB
  - 64bits

3.2. Modelado 3D de la cabeza artificial

Para la primera parte del proyecto, el objetivo es claro: escanear un busto real de tres dimensiones y almacenarlo en formato digital, una malla de puntos que se asemeje a la realidad. El maniquí mostrado en la **figura 21** se ha denominado “Dasha”.

Para la realización del escáner, se utilizan dos métodos, y se aprovechará para compararlos y ver cuál es más preciso.

3.2.1. David LaserScanner

La técnica de obtención de imágenes 3D con David LaserScanner se puede dividir en varias etapas. La primera es colocar el equipo y los accesorios necesarios de la manera adecuada. Estos elementos son: un ordenador con un programa que viene en el pack con la compra del producto y que viene instalado en un pen drive, una cámara web, un láser de color rojo o verde, también dentro del pack y un sistema de paneles para formar un sistema diédrico que servirán para la calibración.

Una vez tenemos todos los elementos, los ubicamos de la manera mostrada en la **figura 23-24**, se coloca la cámara web enfocando al panel diédrico a una distancia tal que entren los puntos característicos marcados en el panel, después se coloca el láser en un ángulo no muy inclinado comparado con el de la cámara. La configuración final se muestra en la **figura 24**.

Figura 23: DAVID 3D LaserScanner

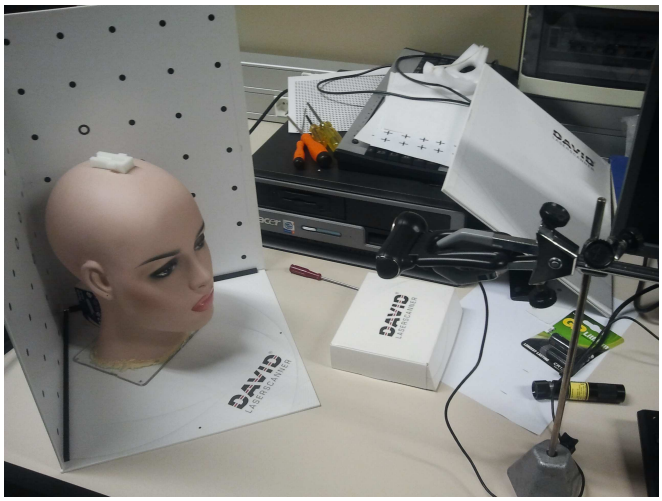


Figura 24: Utilización en el laboratorio de DAVID

Figura 25: Escaneado 3D Dasha con DAVID

Una vez se tienen todos los elementos bien situados, se calibra la cámara con el software, después se pone el objeto en el centro del sistema y, manteniendo las mismas condiciones de iluminación, se van adquiriendo puntos en 3D de diferentes zonas o lados del objeto y guardándolas en archivos.

Por último, se procede al tratado de estas imágenes o "*patches*", y se van entrelazando unas con otras con ayuda del software y eliminando las superficies escaneadas erróneas o el ruido que se produce durante el escaneado manual. El proceso es arduo y costoso en el tiempo, pero los resultados son aceptables. Se observa el resultado con "Dasha" en la **figura 25**.

3.2.2. Konica Minolta Vivid 910

Para el desarrollo del escaneo digital del busto también se ha utilizado otro método. Esta segunda forma incluye un sistema comercial, Konica Minolta Vivid 910, que se encuentra en el campus de Tudela, en la zona de diseño Industrial de la Universidad Pública de Navarra.

En este caso, también hay que colocar los objetos de una manera particular. Primero situamos un soporte a una altura adecuada para que la cámara del sistema esté bien

enfocada hacia el sujeto; después colocamos un fondo negro, ya que ésta máquina se encarga de detectar todos los colores excepto el negro y después conectamos la cámara al ordenador por USB.

Una vez se tiene el conjunto bien fijo, **figura 26**, el procedimiento es bastante parecido al anterior. Vamos tomando imágenes de cada lado o "*patch*" y después utilizamos un software, en este caso PET (Polygon Editing Tool Version 2.21) para ir registrándolas y corrigiéndolas.

Con ésta máquina el resultado es más rápido y más preciso, con menos ruido y con superficies más suaves y fiables. El único problema que tiene es que al ser los ojos de Dasha negros, tiene un rango de error en esa zona. Se corrige pegando unos trozos de cinta aislante en dicha superficie y repitiendo el escaneo en esa región.

Figura 26: Durante Escaneado de Dasha con Konica Minolta

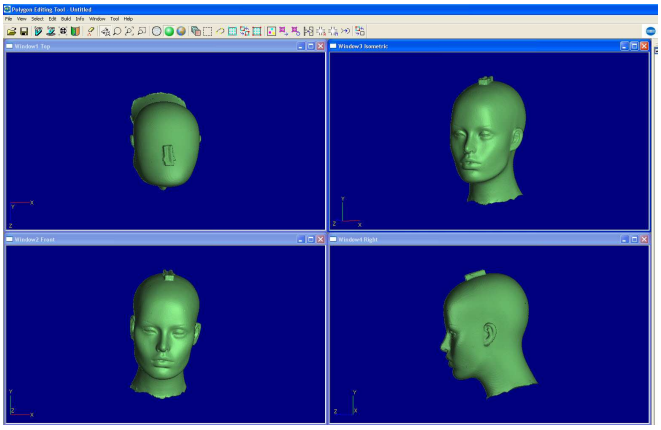


Figura 27: Escaneado 3D Dasha con PET y Konica Minolta

Los resultados se muestran en la **figura 28**. En la izquierda la malla que se obtiene con dicho programa PET y a la derecha una fusión sólido obtenida con el programa de DAVID LaserScanner.

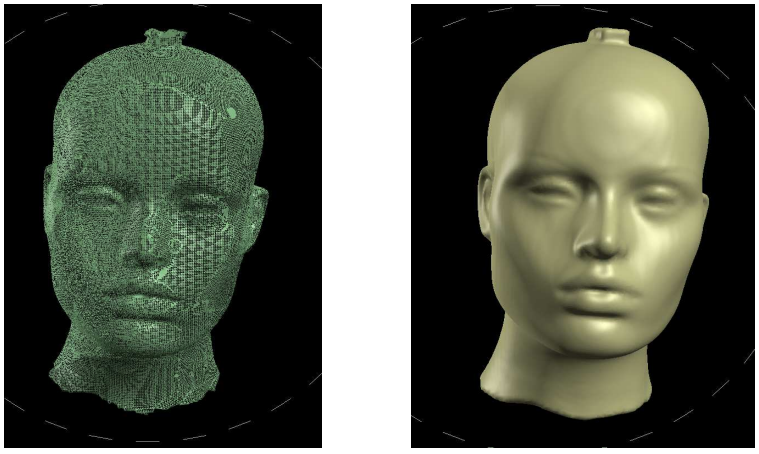


Figura 28: Resultado del Escaneo con Konica Minolta en el Software de DAVID 3D

3.2.3. Comparativa

Habiendo tomado los datos digitales de las dos maneras, se realizará una comparación entre ellas minuciosamente. En la siguiente imagen (**figura 29**) se puede observar cómo es una con respecto de la otra (La morada está tomada con DAVID LaserScanner y la amarilla y la malla de puntos con la Konica Minolta). Se recuerda que las medidas se dan en milímetros para ambos casos.

En la imagen se pueden apreciar algunas diferencias entre ambas, pero está claro que en general la imagen obtenida por la Konica Minolta es mejor: tiene los relieves más marcados (véase labios y orejas) y la textura es mucho más suave. Sin embargo, se pueden establecer más diferencias entre una y otra.

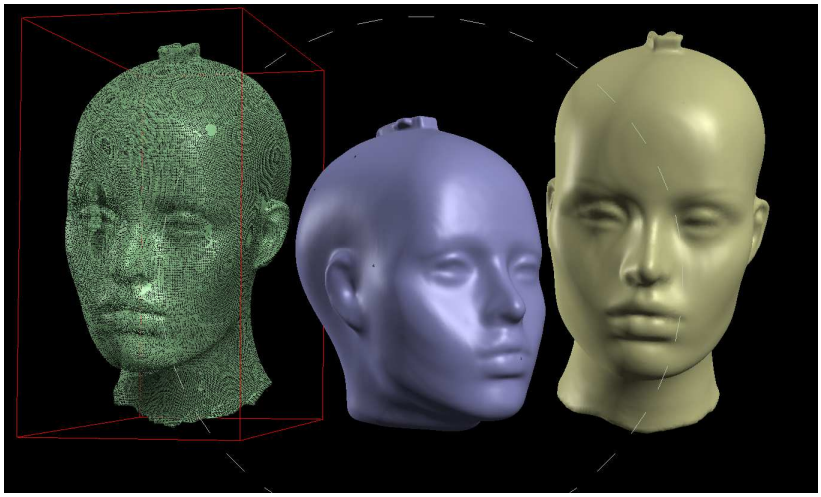


Figura 29: Escaneados Konica (a los lados) y escaneado con DAVID (medio)

Característica	**DAVID LaserScanner**	**KONICA MINOLTAVIVID 910**
Precio	500 €	8.000 €
Tiempo para obtener imagen	5 horas más especialización	3 horas
Calidad de imagen	Buena	Muy buena
Necesidad computacional	Muy alta	Alta

En esta tabla se observa que si bien los resultados son mejores con la Konica Minolta, es vital establecer un compromiso en cuanto al precio, ya que la Konica Minolta cuesta dieciséis veces el DAVID LaserScanner, aproximadamente.

Ahora bien, con DAVID LaserScanner hay que invertir un mayor número de horas, tanto para entender el funcionamiento del programa como para obtener tan buenos resultados, mientras que con la Konica Minolta y un pequeño tutorial de PET de diez minutos se puede llevar a cabo el escaneo digital en pocas horas.

En cuanto a los datos obtenidos, se ha realizado un estudio cuantitativo entre los puntos del modelo escaneado con DAVID LaserScanner y con la Konica Minolta, estableciendo que las Distancias Euclídeas de cada punto de un modelo con respecto al punto más cercano del otro son muy similares.

Estas Distancias Euclídeas se han obtenido cargando ambos modelos en Matlab y realizando un pre-registro entre ellos, un procesado ICP entre las nubes de puntos y un cálculo de Distancias Euclídeas. En el capítulo siguiente que trata de la determinación de los puntos característicos en el modelo se concretará en qué consisten estos procesos. En los histogramas (**figura 30**) se observa que en general las distancias son de un milímetro, la media es para los puntos del modelo Konica con respecto al de David de 2,06 mm y para los puntos del modelo David con respecto al de Konica de 1,13 mm. Se pueden observar también varios puntos que se alejan de los modelos, denominados *outliers*.

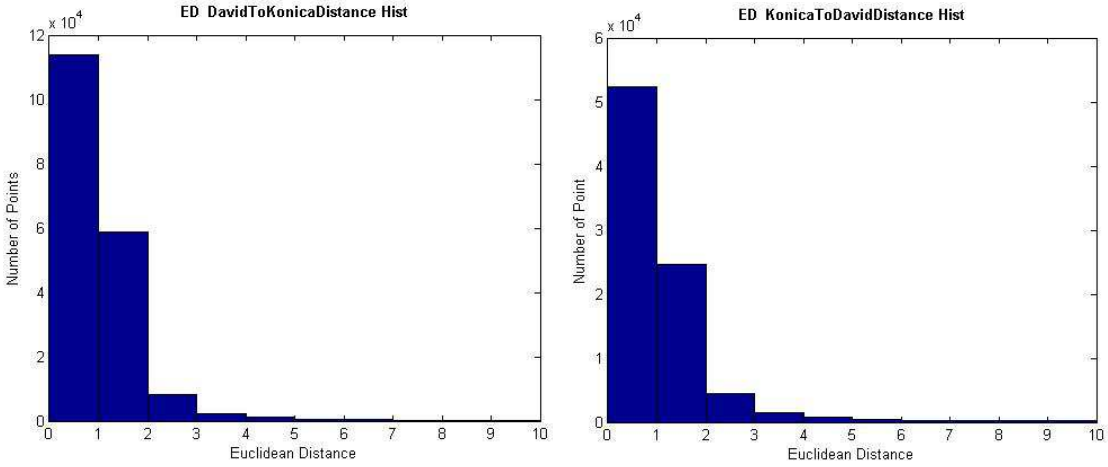


Figura 30: Histogramas de las Distancias Euclídeas entre los modelos digitales de DAVID y Konica Minolta

3.3. Determinar los puntos característicos en el modelo 3D

Hasta ahora, se han obtenido los puntos 3D del modelo de Dasha en formato .OBJ. Se transforman a través de Matlab a un formato .xyz, de tal manera que se obtienen un total de 100.174 puntos con el escaneo de la Konica Minolta.

Sin embargo, para la evaluación de los algoritmos de seguimiento y POSIT en otros proyectos se requerirá comparar con un modelo que remarque los puntos que se están observando y que son característicos. Se trata de 54 marcas particulares de la cara de una persona. Estas 54 marcas se destacan en la **figura 31**.

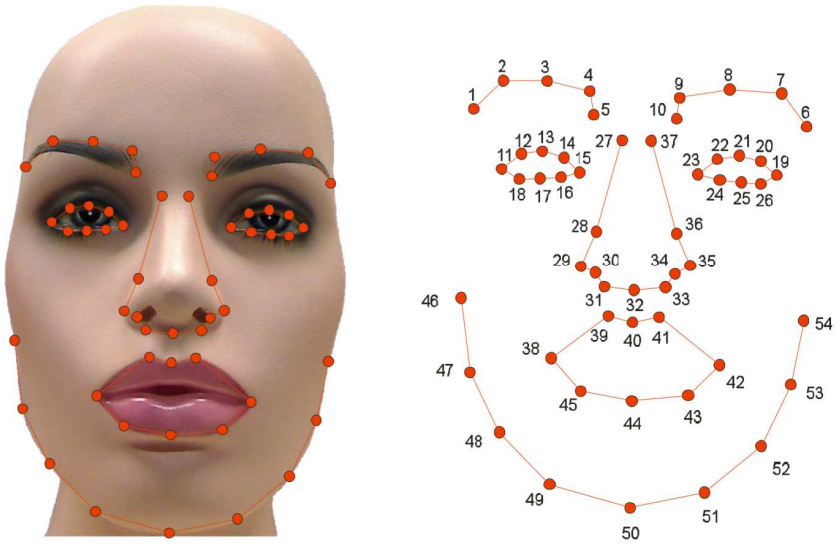


Figura 31: Puntos de Marcado en modelo de Dasha y su indexación [19]

Estos puntos que se marcarán en el modelo son los que se van a evaluar en el seguimiento y en el POSIT. A partir de una serie de comparaciones con ellos se calcula la posición de la cabeza con respecto a la cámara y los correspondientes ángulos de Roll, Yaw y Pitch del usuario a lo largo de los vídeos. Este apartado trata de cómo se han calculado de manera precisa estos 54 puntos en el modelo 3D de Dasha en posición frontal.

Una vez se tienen los puntos del modelo en coordenadas xyz, el siguiente paso es obtener estos 54 puntos característicos. A priori, puede parecer una buena solución abrir el modelo escaneado 3D con Matlab y seleccionar los puntos manualmente. No obstante, esta técnica es bastante imprecisa y puede dar lugar a un gran margen de error, así que se plantea otra solución.

Durante la realización de los vídeos, se han marcado estos 54 puntos con los sensores magnéticos para poder calcular el *GroundTruth* (datos que se suponen reales) de su posición a lo largo de cada una de las grabaciones.

Esto se consiguió de la siguiente manera: en la cabeza del maniquí se creó una pieza[19] para colocar uno de los sensores de tal manera que se moviera solidariamente con la cabeza. A partir de ese origen, con otro sensor se marcaron uno a uno los 54 puntos, y así se obtuvieron las coordenadas de cada punto con respecto del sensor

superpuesto. Después, a través de una serie de procedimientos matemáticos se calcularon las coordenadas en 3D y en 2D de los puntos en cada fotograma de los vídeos grabados.

Así, como para cada fotograma se conocen estos puntos marcados con el sensor magnético, se puede utilizar cualquiera de estos datos para tener una relación en el espacio entre ellos y cargarlo en Matlab.

Se crea un programa llamado "*FindingDashaPoints.m*" cuyo objetivo es realizar un registro entre el modelo 3D escaneado y los puntos faciales marcados con el sensor. En este programa, se procede como sigue: primero se cargan los puntos del modelo. Estos puntos no están en posición totalmente frontal, y es necesario que sea así para poder compararlo con el POSIT, así que se rotan para colocarlos en posición frontal. Después se traslada el centro de coordenadas al centro del busto y se guarda este modelo 3D. Este es el primer dato que se obtiene importante para siguientes proyectos: el modelo "Dasha" en tres dimensiones perfectamente centrado y en una posición frontal.

El siguiente paso consiste en cargar los 54 puntos obtenidos con el sensor, pero al estar referidos al origen del propio sensor, su sistema de coordenadas con respecto al modelo 3D de Dasha es diferente. Como solución se realizan una serie de cálculos. Inicialmente se rotan los ejes del sistema de coordenadas para que coincidan con los de Dasha. A través de los datos de las grabaciones y del modelo, se obtiene la rotación axial correspondiente:

$$x' = -y;$$
$$y' = z;$$
$$z' = -x;$$

También hay que corregir el hecho de que la base del sensor no esté bien alineada con los ejes naturales de la cabeza (se puede observar en la **figura 21**). Esta matriz de rotación está ya calculada y es:

$$R = \begin{pmatrix} 0{,}9929 & -0{,}00073 & -0{,}1186 \\ -0{,}0037 & 0{,}9993 & -0{,}0374 \\ 0{,}1185 & 0{,}0375 & 0{,}9922 \end{pmatrix}$$

Ahora llegan una serie de procesos para calcular los 54 puntos más cercanos al modelo. El primero es trasladar el modelo 3D al mismo origen que los puntos del

sensor. Se halla la translación aproximada manualmente (luego se precisará mejor) y consiste en restar a los datos el siguiente vector de posición:

$$\begin{bmatrix} -3{,}039 & -164{,}5 & 20{,}4 \end{bmatrix}$$

Hasta ahora, todo este procesado se denomina pre-registro, que consiste en alinear lo mejor posible y de manera "manual" el sistema de coordenadas del sensor y el del modelo escaneado, y se hace de manera previa al registro.

Existen programas ya creados en Matlab que realizan la función de Acercamiento al Punto más Cercano (*Iterative Closest Pointo o ICP*) que implementan un algoritmo para minimizar la diferencia entre dos nubes de puntos y acercarlas entre sí.

Para este paso, se han utilizado dos algoritmos implementados [10] y se ha evaluado cuál es el mejor calculando las Distancias Euclídeas entre los puntos de salida del algoritmo y los puntos más cercanos a ellos del modelo de Dasha en 3D.

Después de la evaluación, se ha utilizado la matriz de rotación y traslación de salida del algoritmo ICP y se han rotado y trasladado los puntos, para que queden lo más cerca posible de la superficie del modelo. La colocación entre el modelo y los puntos puede verse en la **figura 32**.

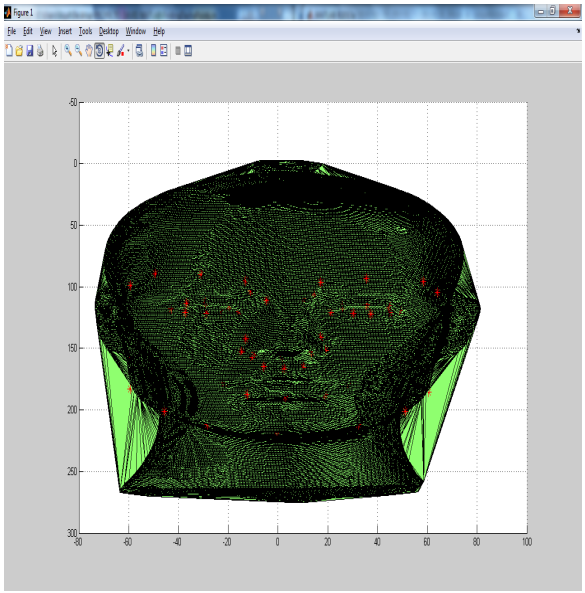


Figura 32: Registro de puntos en Dasha

Por último, estos puntos tienen como origen el centro del sensor, pero para el modelo de Dasha esto no nos sirve. Volvemos a trasladar el origen al centro de la cabeza, a la altura de la nariz y justo en el lóbulo de las orejas. Estos puntos ahora están lo más cercanos posible después de hacer el registro, pero aun así no forman parte exactamente del modelo de Dasha. Para algunas otras aplicaciones, necesitamos exactamente los puntos en Dasha.

Por ello, la última parte de todo el programa de registro realiza una matriz de Distancias Euclídeas de cada punto con el resto de puntos del modelo, y se queda con la distancia más corta. Este punto luego es el que se coge en el modelo de Dasha. De aquí sacamos varios resultados que nos dan valores sobre la calidad del registro. La distancia más larga de entre los puntos se localiza en la mandíbula izquierda en el dibujo

(se puede observar que está fuera de Dasha ligeramente) y tiene un valor de 1.7 mm, lo cual es un error lo suficientemente pequeño como para permitir su uso. También se calcula que la media de las distancias es de 0.71 mm.

Después, se guardan los puntos en diferentes .mat y .txt para utilizarlos en la evaluación de diferentes algoritmos.

3.4. Bases de datos de entrada

En esta sección se comienza con el bloque de análisis de algoritmos para seguimiento de objetos y su evaluación con el POSIT. La mayor parte de las pruebas se van a realizar en dos bases de datos muy diferentes entre sí, una de la *Boston University (BU)* y la otra de la Universidad Pública de Navarra (UPNA). Ambas son bases de datos de secuencias de vídeo en las que los usuarios realizan distintos movimientos de la cabeza delante de la cámara.

Es necesario explicar brevemente en qué consiste cada una y sus peculiaridades. En el caso de la base de vídeos de la UPNA porque se ha colaborado en su realización y en el caso de la BU porque tiene algunas características especiales en el momento de implementar programas con ella.

3.4.1. Universidad Pública de Navarra

En un principio, la base de vídeos de la UPNA es con la que se ha trabajado en todos los casos de algoritmos implementados. Se ha creado a partir de la colaboración del proyecto de José Javier Bengoechea[19] y Rebeca Echeverría[20] . En esta base de datos se grabó un vídeo con el maniquí “Dasha”, después José Javier grabó un total de diez usuarios y estos se han utilizado para este proyecto.

Cada set de 12 vídeos por usuario tiene su propia calibración, y por tanto cada vez que se utiliza en la implementación hay que tener en cuenta dónde está la distancia focal y el centro de la imagen para la ejecución del POSIT. Aunque ambos son parámetros intrínsecos a la cámara web, para cada usuario la calibración es distinta, es decir, cada sistema completo (cámara-transmisor-sensores) se calibran antes de cada grabación para calcular las matrices de transformación entre ellos.

También cabe destacar que los vídeos tienen mejor calidad y mayor resolución que los de la BU, y por ello en general los resultados deberían ser más precisos, teniendo en cuenta que también ocuparán más tiempo en el procesado como espacio en el disco.

3.4.1.1. Calibración del sistema

Para la calibración del sistema, existen varios componentes para graduar y medir, tales como la cámara web, el transmisor magnético, el receptor magnético y el objeto en cuestión.

Por sistema, la calibración se realiza en dos líneas diferentes, la primera es la calibración de la cámara con el plano en el que se van a grabar los vídeos, y la segunda es la calibración del transmisor magnético con su sensor. Después, con estos datos se calculan una serie de matrices que indican la transformación de unos sistemas a otros para tener el sistema global.

En la consecución de la calibración, se ha de destacar que todos los algoritmos y procedimientos estaban ya implementados por los dos proyectos mencionados anteriormente.

Por claridad, se procederá a explicar los principios en los que está basado el método para la grabación de los vídeos de la biblioteca de UPNA.

3.4.1.1.1. Calibración de la cámara con el entorno real

La calibración de la cámara se lleva a cabo mediante el Camera Calibration *Toolbox for Matlab de Jean-Yves Bouguet*. Este toolbox se encarga de hallar, a partir de una serie de imágenes, los parámetros intrínsecos y extrínsecos de la cámara.

Para la biblioteca de la UPNA, la calibración se realiza capturando un total de 50 imágenes con la cámara fija (estableciendo un círculo de seguridad para que nadie la mueva y sujetándola sobre un soporte recio y robusto). En estas imágenes se fotografía la rejilla que indica el *toolbox* en diferentes posiciones con respecto a los ejes X, Y y Z, y con diferentes rotaciones sobre el trípode en el que se sujeta.

Con las mejoras del *toolbox*, no hace falta marcar nada más en cada rejilla, con estas imágenes ya se dispone de toda la información.

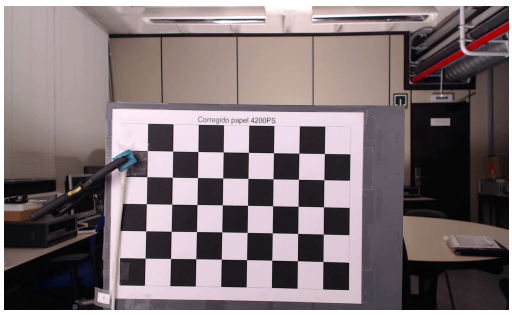


Figura 33: Damero con el que se realiza la calibración

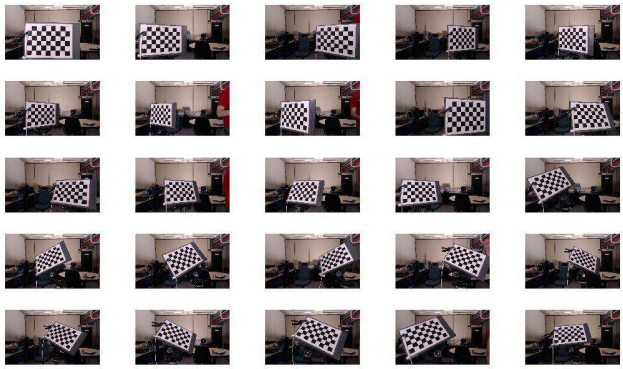


Figura 34: 25 de las 50 imágenes con las que se calibra la cámara y el sistema

Es necesario añadir que si el número de imágenes tomadas llega a 50 se consigue el mejor resultado en cuanto a precisión, y ésta es un punto clave del proyecto ya que a partir de estos datos basaremos todos los resultados, tomando como *GroundTruth* los datos obtenidos gracias a este sistema de medida.

Con este *Toolbox* se consiguen datos intrínsecos tales como la distancia focal y el punto principal (que luego se utilizarán para el POSIT) y por otro lado se obtienen parámetros extrínsecos como la rotación y traslación del origen de cada plano que contiene a la rejilla con respecto al sistema de referencia de la cámara.

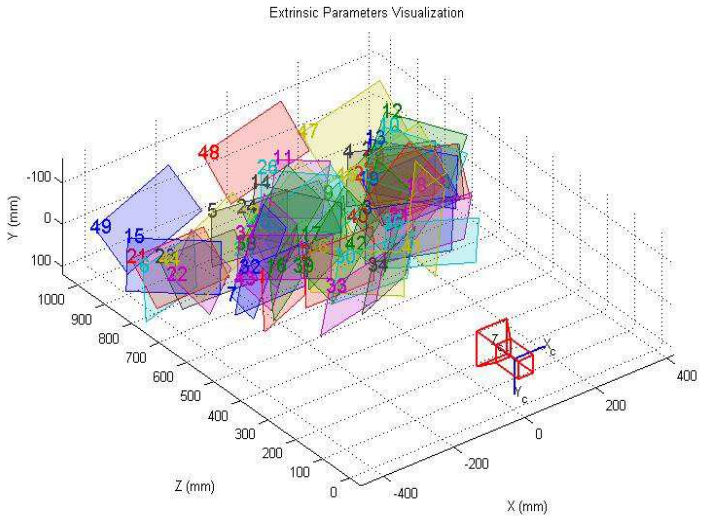


Figura 35: Conjunto de posiciones en las que se fotografía el damero en las 50 imágenes

Una vez tenemos calibrada la cámara, ya se tiene la relación existente entre la cámara y el plano o rejilla en el que se van a grabar los vídeos.

3.4.1.1.2. Calibración del sensor magnético

Para la elaboración del *GroundTruth* se necesita conseguir referenciar el sensor magnético con respecto de la cámara con la que se van a grabar los vídeos. En este apartado se estudiará cómo se trabaja con el sensor.

El sensor tiene un transmisor y dos receptores. El transmisor está colgado del techo en un punto fijo, y es el receptor con el que se va a jugar, ya que está conectado por cable al centro de operaciones del sensor y nos permite bastante movilidad. Uno de los receptores se pone, a la vez que se hace la calibración de la cámara, pegado a la rejilla en uno de los puntos, como se puede observar en la **figura 36**.

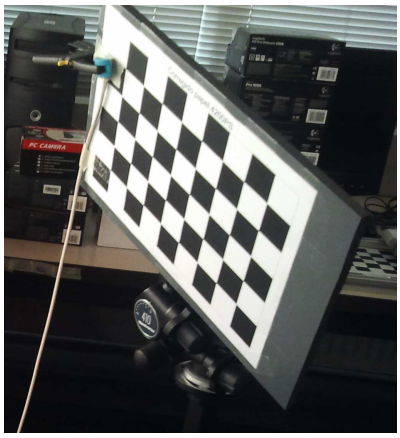


Figura 36: Damero con el que se realiza la calibración y el sensor acoplado

El propio sensor tiene una serie de aplicaciones que permiten trabajar con él. La aplicación que se utiliza es *GetSynchronousRecord*, la cual obtiene archivos .dat en los que se guardan varios datos referentes al número de sensores: la traslación de cada uno de éstos en los ejes X, Y y Z en pulgadas, el vector de rotación, el Azimuth, la elevación y Roll con respecto al transmisor.

Cada vez que se toma una imagen de calibración se promedian los datos que indica el sensor, de tal manera que se registran las posiciones del sensor respecto al transmisor una vez está fijo en el plano de la rejilla. Hay que destacar que cuánto más cerca está el receptor del transmisor más precisos son los datos obtenidos.

Gracias a las funciones creadas por los dos compañeros el proceso se hace simultáneo y de manera bastante sencilla.

Proceso de optimización:

Por último, en el sistema de calibración del sensor se deben mencionar dos procesos de optimización que se llevaron a cabo por mis compañeros. El primero es la obtención del origen del sensor.

El sensor de posición electromagnético se encuentra encapsulado dentro del receptor. Las dimensiones de éste son de 7.9x19.7x7.9 mm, pero no se dispone de la localización del centro magnético, por lo que Rebeca Echeverria[20] en su momento se encargó de calcular el origen exacto para que las medidas fueran más exactas.

La otra optimización se debe a cómo está asociado el receptor a la rejilla y a la medición de la matriz desde un sistema a otro. Hay que recordar que este sensor está pegado con celo en una de las esquinas. De esta manera, como se mantiene la posición entre el receptor y la rejilla fija en todas las imágenes, se puede calcular la matriz entre el receptor y el plano M12 de manera manual (**figura 36**).

No obstante, puesto que este proceso de pegado es manual y por tanto impreciso, José Javier[19] se encargó de mejorar la medida de colocación del sensor de tal manera que se optimiza cada proceso de calibración para minimizar los errores, asociando los puntos de la rejilla con el sensor y calculando a partir de una función la propia rejilla, obteniendo la matriz con la que las medidas salen más cercanas a la realidad.

3.4.1.1.3. Cambio de coordenadas entre los sistemas

Hasta ahora se han conseguido los datos entre cámara-plano de los vídeos y entre sensor transmisor-receptor. También se necesita hallar una transformación desde el receptor hasta la cámara, de tal manera que se puedan obtener los resultados del *GroundTruth*. Para esto hay que realizar una serie de cálculos transformando matrices.

Para tener los conceptos más claros, vamos a enumerar los cuatro sistemas de los que disponemos. Al transmisor magnético se le denomina sistema 0, al receptor el sistema 1, al plano donde se halla la rejilla el sistema 2 y a la cámara el sistema 3. En la **figura 37** se puede ver cómo es el sistema global.

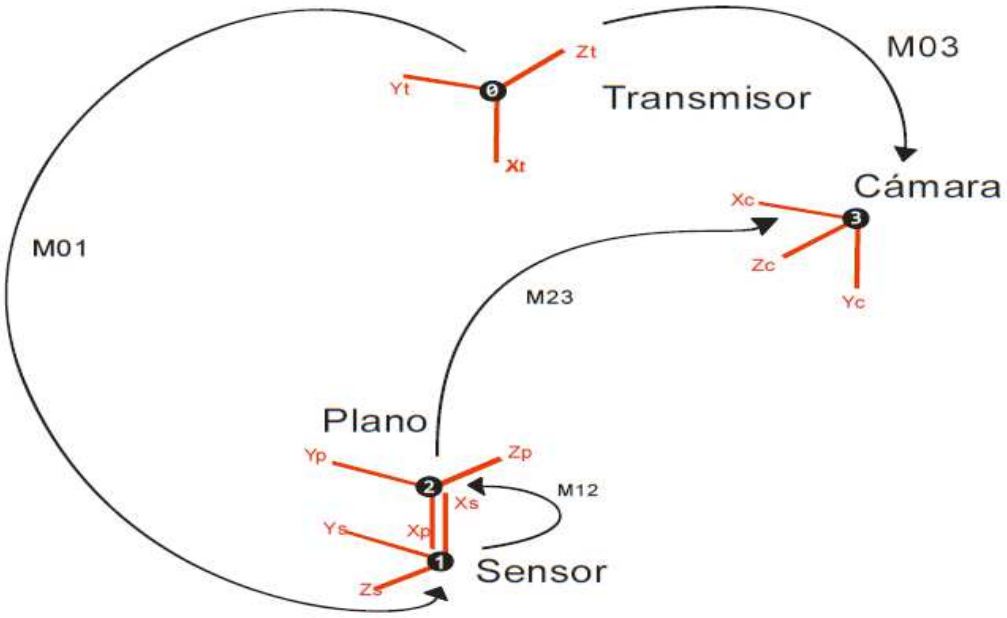


Figura 37: Sistema de coordenadas entre los diferentes elementos para la base de vídeos.

El objetivo principal de este apartado es encontrar la manera de trasformar desde el sistema 0 (transmisor) hasta el sistema 3 (cámara). Para encontrar esta conversión se utiliza el método de la matriz de transformación homogénea, la cual transforma la rotación y traslación de un punto en su sistema de referencia al mismo en otro sistema de referencia. Una vez se tenga la matriz transformación de los puntos, será posible obtener a partir de los datos del transmisor-receptor la posición de los puntos con respecto a la cámara.

La matriz de Transformación Homogénea es una matriz 4x4 en la que se incluyen el vector de traslación y la matriz de rotación, y se añade una fila de ceros con un uno al final, de tal manera que permite, dado un punto en el sistema de coordenadas inicial, transformarlo a sus coordenadas en el sistema final. La forma de esta matriz sería:

$$MTH = \left(\begin{array}{ccc|c} n_x & s_x & a_x & t_x \\ n_y & s_y & a_y & t_y \\ n_z & s_z & a_z & t_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

donde la matriz (N,S,A) = 3x3 es la matriz de rotación y la T = 3x1 es la de traslación. De igual manera, la matriz de transformación en sentido inverso será:

$$MTH^{-1} = \left(\begin{array}{ccc|c} n_x & n_y & n_z & -N \cdot T \\ s_x & s_y & s_z & -S \cdot T \\ a_x & a_y & a_z & -A \cdot T \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

donde N = (n_x n_y n_z), S = (s_x s_y s_z), A = (a_x a_y a_z), y T es el vector traslación.

Con todo lo que se ha ido comentando en los apartados anteriores, se dispone de las matrices de transformación siguientes:

- Transmisor-receptor sensor = M01 / M10
- Cámara-plano de referencia = M32 / M23
- Plano-receptor = M12 / M21

De esta manera, con las funciones diseñadas por los compañeros se obtiene una matriz de transformación entre el transmisor y la cámara que relaciona todo el sistema de la **figura 37**:

$$M03 = (M23 * M12 * M01)$$

Es muy importante mantener el orden ya que al multiplicar matrices de rotación el orden de los factores altera el resultado. También se puede obtener la matriz de transformación inversa, es decir, entre la cámara y el transmisor que será:

$$M30 = (M10 * M21 * M32)$$

Para cada imagen de las 50 tomadas en la calibración se calculan estas matrices, y la resultante, que es la que se va a utilizar en las grabaciones, se obtiene promediando todas ellas.

3.4.1.2. Marcaje de los puntos característicos de los usuarios

Como ya se ha explicado, es necesario realizar un marcaje de los puntos característicos de los usuarios para poder referenciarlos y saber sus posiciones con

respecto al sensor que se sitúa en el soporte de la parte de arriba en el caso de Dasha o en una diadema con una pieza especial en el caso de los usuarios reales.

Para este marcado, se utilizan varios programas ya implementados para calcular la posición de los puntos. Se irá marcando punto a punto los que nos interesan, los 54 puntos señalados anteriormente. En la **figura 38** se puede observar cómo se marcaron en el usuario "Dasha" con una pieza especial en la que se apoya uno de los receptores magnéticos:

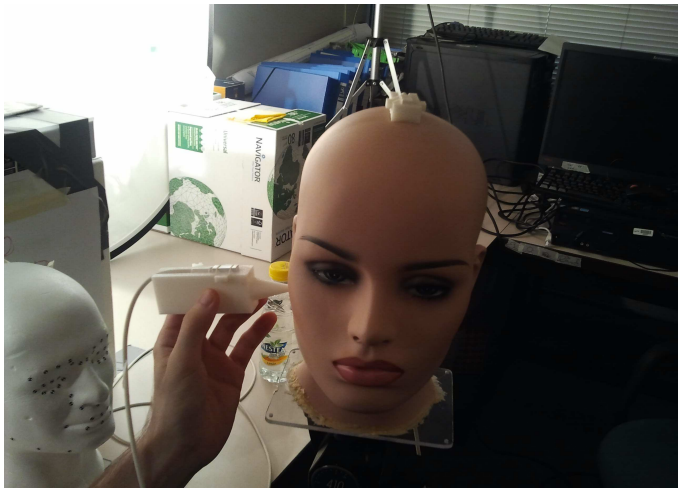


Figura 38: Marcado de los puntos característicos de Dasha con el

3.4.1.3. Elaboración de vídeos

En el apartado anterior se ha expuesto el proceso de calibración para la grabación de la base de datos de la UPNA. En esta sección se va a comentar cómo serán los vídeos grabados. Estos vídeos servirán luego para aplicar los algoritmos de seguimiento sobre ellos y compararlos con el *GroundTruth*, y después ver la influencia de este seguimiento sobre la estimación de la posición y rotación de la cabeza calculada.

En cada vídeo, el usuario lleva una diadema con el sensor anclado a ella. En el caso de Dasha se creó su propio soporte, pero en todos los casos el sensor está en la cabeza del usuario y se mueve solidariamente con ella. Ahora hay que determinar qué tipo de vídeos nos interesan. Para esta base de datos, se ha decidido que los vídeos durarán diez segundos y tendrán los seis primeros unos movimientos guiados (se les indica a los usuarios en qué direcciones) y los seis últimos movimientos libres en los que el usuario decide.

En los seis primeros se va a tratar de hacer los seis movimientos que definen a una persona en frente de una cámara web. Estos son, en el mismo orden que se van a grabar

los vídeos: traslación horizontal de izquierda a derecha, traslación vertical, traslación horizontal de delante hacia atrás, rotación de la cabeza en Roll, rotación en Yaw y rotación en Pitch.

Estos últimos giros quedan definidos si el sujeto se coloca con los ojos mirando hacia la pantalla como si fuera el eje x apuntando a la pantalla en la **figura 39**.

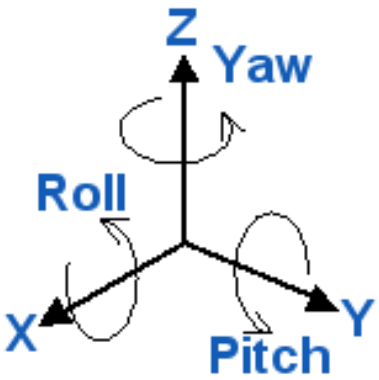


Figura 39: Rotaciones que puede tener la cabeza humana

En cambio, en los seis vídeos posteriores, como se ha dicho, se graban movimientos libres, cada usuario puede rotar y trasladar su cabeza a su antojo. Así, estos últimos simulan un conjunto de movimientos aleatorios a lo largo de los diez segundos, más fiel a la realidad.

Por último, los vídeos se graban cada uno con 300 fotogramas y con una resolución de 1280x720 píxeles por fotograma, después esos vídeos se procesan y se obtienen los datos de pose y las proyecciones de los puntos marcados en 3D en cada fotograma. Con esto se genera un nuevo vídeo en el que se representan ambos grupos de datos: los puntos seguidos y el *GroundTruth* de la pose.

3.4.2. Boston University (BU)

Con respecto a la Universidad de Boston, hay que tener en cuenta una serie de características diferentes con respecto a la base de datos de la UPNA. La primera es que su resolución es más pequeña (320x240 píxeles), lo que puede dar lugar a menos precisión en los resultados.

Además, no se tiene el *GroundTruth* para el seguimiento de los puntos, así que en este caso nunca se calcula. Tampoco se tiene la calibración de la cámara con la que se

grabaron los vídeos, por tanto desconocemos el centro óptico y la distancia focal. Sin embargo, experimentalmente se ha calculado que la distancia focal es la siguiente:

$$[\,485 \quad 485\,]$$

Y como centro óptico se toma el centro de la imagen:

$$[\,160 \quad 120\,]$$

Por último, también se debe mencionar que en los vídeos de la BU hay que realizar un *flip* de las imágenes respecto al eje Y para todos los vídeos, porque los resultados están dados para un sistema de coordenadas en el cual el eje X está invertido. Para hacerse una idea, el resultado de hacer *flip* a una imagen se observa en la **figura 40**.

Figura 40: Demostración de cómo cambia el fotograma con *flip*. A la izda. el fotograma no tiene *flip* y a la derecha sí .

3.5. Evaluación y análisis de algoritmos de seguimiento

Esta es la parte más extensa del proyecto. Consiste en utilizar los videos que se han grabado y en los cuales se tiene una serie de datos que se consideran verdaderos (*GroundTruth*) para evaluar una serie de algoritmos y dilucidar cuál da mejores resultados.

3.5.1. Marcaje de puntos de interés en primera imagen de vídeo

Inicialmente, para la UPNA se tienen doce vídeos de cada usuario que serán los que servirán para evaluar los algoritmos. Sin embargo, lo primero que hay que tomar son los puntos de interés en el primer fotograma, esto es, los puntos que posteriormente se van a introducir en el algoritmo para que se calcule su seguimiento a lo largo del vídeo.

En el caso de la BU, en la propia base de datos se encuentran los archivos con las coordenadas de los 12 puntos iniciales, éstos puntos característicos se pueden observar en la **figura 41**. Aunque "Dasha" no forma parte de esta base de datos, los puntos marcados son los mismos y la calidad de la imagen es mejor.

En el caso de la UPNA, hay que definir qué puntos se van a introducir. En este caso se marcarán 54 puntos, siguiendo un patrón dado en la **figura 41**. El objetivo de tener 54 puntos es calcular si hay diferentes resultados con más puntos o no y la dependencia y robustez de cada punto con los algoritmos de seguimiento.

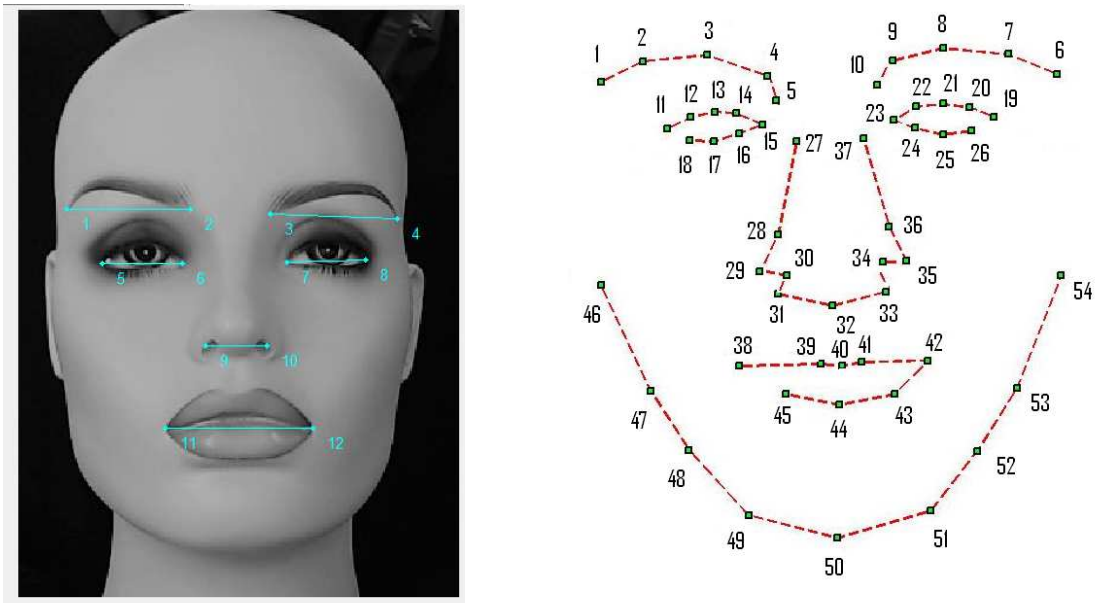


Figura 41: Dasha con los 12 puntos característicos y los 54 puntos en forma de cara

Con este marcado de 54 puntos, los 12 puntos de la base de vídeos de la BU están incluidos entre ellos. Cuando se requiera utilizar sólo los 12, se copiarán los 12 puntos de interés en otro archivo, de tal manera que con el mismo marcado se obtienen los dos grupos de puntos.

Para el marcaje, se utiliza un programa proporcionado por la Universidad Pública de Navarra, llamado *"Landmarks_Definition_App_(Matlab_GUI)"* el cuál está preparado para marcar los puntos con alta precisión y guardar sus coordenadas.

Para ello, se necesita la imagen correspondiente al fotograma inicial de cada vídeo. Se crea otro archivo de Matlab llamado "*OutputFirstImage.mat*" y se realiza el mismo proceso cargando los 120 vídeos, y guardando las primeras imágenes en diferentes archivos.

Una vez hecho esto, se abre la herramienta de "*Landmarks*" con Matlab, (dispone de guía en la propia carpeta adjuntada), y se van cargando cada una de las imágenes y apuntando las marcas que necesitamos. Una imagen en la que se puede ver el proceso es la **figura 42**.

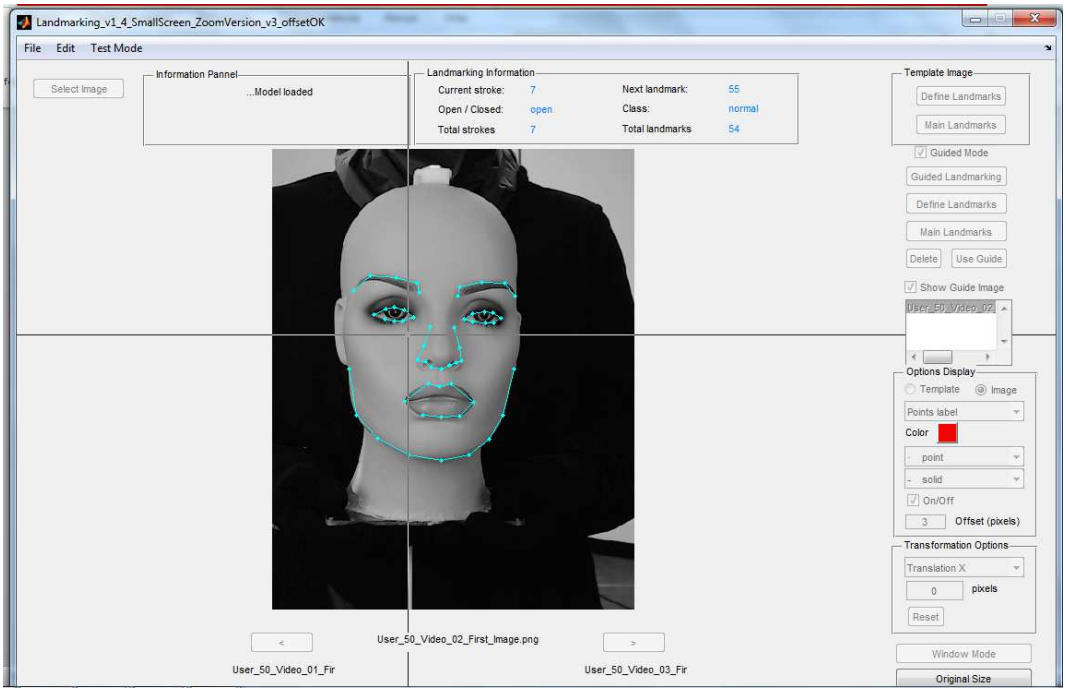


Figura 42: Los 54 puntos marcados con el programa de Landmarking de la UPNA

Una vez se han hallado los puntos marcados con este programa, se guardan en un formato .mat que los agrupa en diferentes tipos de datos. Congrega los datos por subgrupos de líneas (uno por cada una de las cejas, otros dos por los ojos, otro por la nariz, la boca y un último por la barbilla) y en cada subgrupo tiene los diferentes puntos. Por ejemplo, en la imagen hay 7 subgrupos, y el primer subgrupo está formado por 5 puntos, el segundo también, el tercero por 8, y así sucesivamente.

En los algoritmos que se implementarán, interesa introducir los puntos tal cual, sin importar si pertenecen a un subgrupo o no, de tal manera que se tiene que convertir estos datos .mat a otros ficheros en los cuáles estén anotados los puntos de manera independiente.

Para ello se crea el programa "ExtractionPointsFromLandmark.m" en el que se le introduce el .mat que se va a convertir y los nombres de los ficheros que se van a crear y desarrollar archivos de texto. Una vez se hallan los archivos para 54 puntos, de ahí se extraerán los 12 puntos interesantes cuando sea preciso.

Ahora se pueden utilizar estos ficheros .txt y marcar en MATLAB dónde quedarían los puntos de interés, en los cuales se va a realizar el seguimiento, en la primera imagen. Se puede observar el resultado de las imágenes del último vídeo de Dasha en la **figura 43**.

Figura 43: Fotogramas iniciales con los 54 y 12 puntos marcados

De esta manera, se obtienen los 120 archivos de .txt en los cuáles se tiene la posición en píxeles de los puntos que interesan de cada uno de los 12 videos por cada uno de los diez usuarios. En la **figura 44** se representan los primeros fotogramas del primer vídeo de cada usuario.

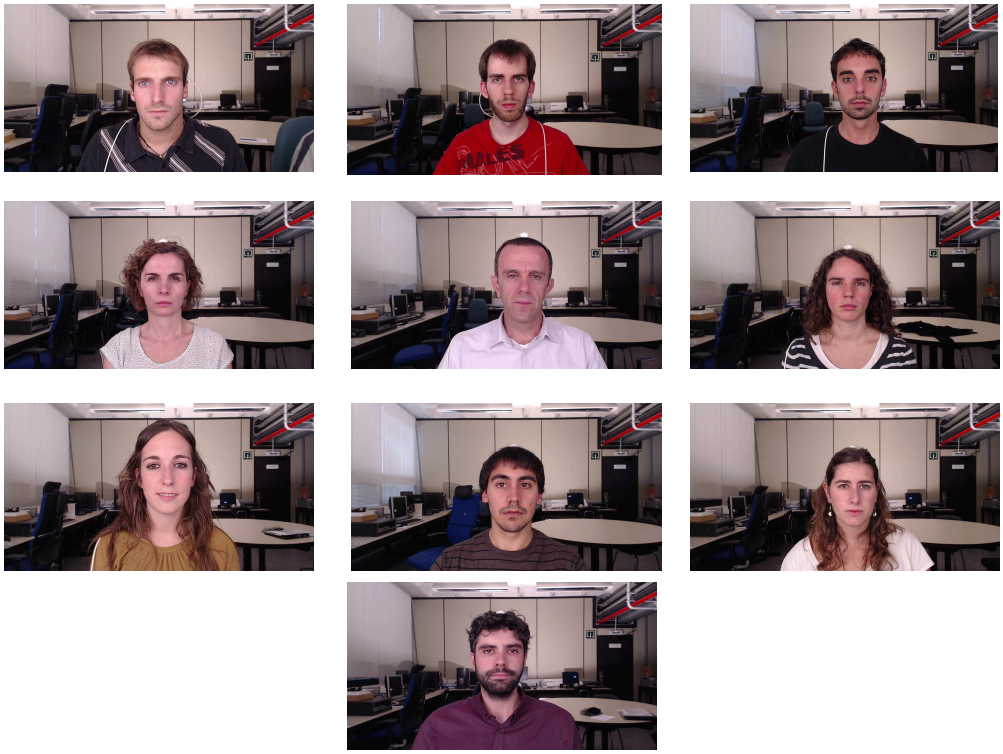


Figura 44: Fotogramas iniciales de un vídeo de cada usuario

3.5.2. Programa de desarrollo de errores de Seguimiento y POSIT

En cualquiera de los casos que se van a trabajar, siempre se cuenta con un algoritmo de seguimiento del que se hablará más adelante (diferentes métodos) que tiene como resultado la salida de los nuevos puntos para el siguiente fotograma en píxeles (en coordenadas x e y). También se obtiene como salida la validez de ese punto, de tal manera que 0 significa que ha dejado de seguirse y ya no es válido y 1 significa que el algoritmo todavía lo considera activo.

A partir de ahí, en este proyecto se van a conseguir diferentes tipos de error para cuantificar los resultados de la evaluación de los algoritmos. El primero, en caso de que la base de datos lo permita (en la UPNA sí, en la de la Universidad de Boston no) es el error de seguimiento. El segundo es el de estimación de la posición y orientación de la cabeza.

3.5.2.1. Error de Seguimiento

Para calcular el error de seguimiento de los puntos, se procede como sigue. Primero, se cargan los valores del seguimiento que se tiene de la Base de Datos y que se conocen como *GroundTruth*. Estos valores indican la posición de los puntos marcados durante la calibración en cada una de los fotogramas de los vídeos.

No obstante, hay que tener en cuenta que los puntos marcados por el útil no coinciden exactamente con los puntos iniciales marcados para cada vídeo manualmente, de tal manera que se parte con un error inicial.

En un primer caso, se pensó en calcular el error total y después restarle el error inicial entre los puntos del *GroundTruth* y los puntos marcados a mano en cada fotograma inicial. Sin embargo, se acabó desechando porque la variación en cada fotograma depende de la rotación y no tenía sentido quitar el error inicial, había casos en los que incluso podía dar un error negativo.

Por ello, se tomó la decisión de hacer una triangulación de los puntos. Este proceso es correcto siempre y cuando se cumpla una característica en los vídeos, que no haya deformaciones de los puntos de la cara (es decir, siempre se mantengan los triángulos iniciales sin grandes variaciones). En nuestro caso es cierto para el *GroundTruth*, ya que los puntos van marcados con coordenadas respecto a un sensor que nunca se deforma.

Así, en la obtención del "nuevo" *GroundTruth* para cada vídeo, primero se hace una relación de triangulación a partir de coordenadas baricéntricas. Esto es, decidir a qué

conjunto de tres puntos del antiguo *GroundTruth* va a pertenecer nuestro punto marcado manualmente. La asignación de coordenadas baricéntricas es correcta siempre y cuando todos los puntos pertenezcan a un mismo plano. En este caso, en la cara, se realiza esta aproximación y se considera el error resultante asumible.

En resumen, para el cálculo de las coordenadas baricéntricas se debe referenciar cada punto que se va a reasignar a otros tres puntos que forman el triángulo en el *GroundTruth* inicial. Así se establece una relación entre los puntos desplazados y los puntos hallados como *GroundTruth*.

En la elección de los puntos y sus relaciones, se probaron varias combinaciones y al final se decidió mantener el siguiente criterio: uno de los tres puntos tiene que ser el correspondiente en el *GroundTruth* y los otros dos son los más cercanos siempre y cuando sean estables y no estén muy juntos entre sí (en casi todos los casos excepto en algunos puntos de la nariz).

En la siguiente tabla está la relación entre los puntos para calcular las coordenadas baricéntricas[21], es decir, cada punto marcado con el útil se va a triangular con otros tres y con la referencia de dónde está nuestro punto en el primer fotograma.

Punto Marcado	Índices de puntos antiguo *GroundTruth* con los que se va a triangular			Punto Marcado	Índices de puntos antiguo *GroundTruth* con los que se va a triangular		
	A	B	C		A	B	C
1	1	2	11	**28**	28	27	37
2	1	2	11	**29**	29	27	37
3	2	12	3	**30**	30	27	37
4	3	14	4	**31**	31	27	37
5	4	15	5	**32**	32	27	37
6	7	6	19	**33**	33	27	37
7	7	20	8	**34**	34	27	37
8	7	21	9	**35**	35	27	37
9	8	22	9	**36**	36	27	37
10	9	23	10	**37**	37	23	36
11	11	12	18	**38**	38	39	45
12	12	13	17	**39**	31	32	39
13	13	17	14	**40**	32	40	41
14	14	15	16	**41**	32	40	41
15	14	16	15	**42**	41	43	42
16	14	15	16	**43**	41	42	43
17	13	16	17	**44**	40	44	43
18	13	18	17	**45**	39	45	44
19	20	26	19	**46**	46	29	47
20	20	26	19	**47**	46	47	38
21	20	21	25	**48**	48	47	38
22	22	23	24	**49**	48	49	45
23	22	23	24	**50**	49	50	44
24	22	23	24	**51**	43	50	51
25	21	20	25	**52**	42	52	53
26	20	26	19	**53**	53	42	54
27	27	5	15	**54**	54	53	42

Por lo tanto, al principio de cada vídeo se calcula el nuevo *GroundTruth* triangulando con este método. Se explicará para un punto y es análogo para los demás.

Primero se toma el fotograma inicial, de este fotograma tenemos los puntos del *GroundTruth* y los puntos marcados a mano. Se puede observar en la imagen el *GroundTruth* en azul y los puntos marcados a mano en verde en la **figura 45**.

Con el fotograma inicial se puede realizar el cómputo del nuevo *GroundTruth*. Primero se calculan las dos lambdas que describen la posición del punto con respecto a los otros tres. Estas dos lambdas son las coordenadas baricéntricas.

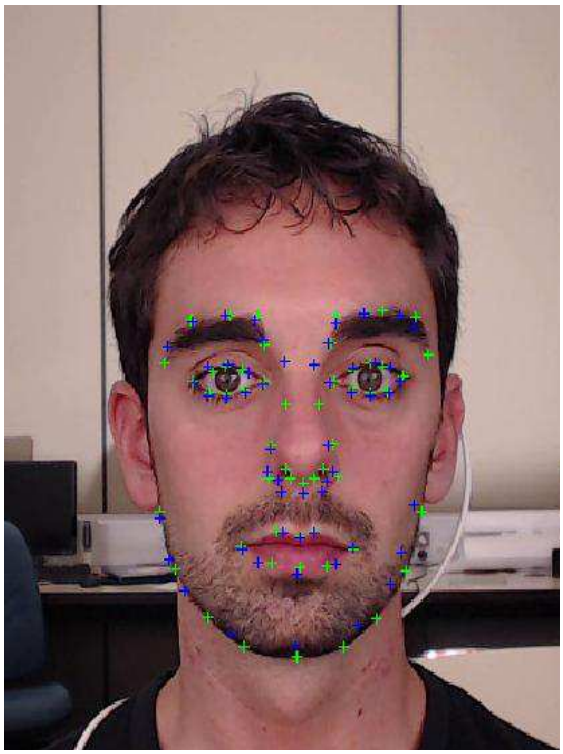


Figura 45: Usuario con los 54 puntos del GroundTruth en azul y los puntos marcados manualmente en verde

Supongamos el punto 1 marcado (A), sus puntos relativos son el 1 (r), 2 (s) y el 11 (t). Entonces calculamos sus lambdas así:

$$Ai = \begin{pmatrix} r_x - t_x & s_x - t_x \\ r_y - t_y & s_y - t_y \end{pmatrix}$$

$$\Lambda_i = (A_i)^{-1} * (A(x,y) - t(x,y))$$

Así obtenemos las lambdas (Λ_i) del punto 1, y esto nos describe su situación con respecto de los otros tres. Ahora lo único que tenemos que hacer en cada escena es calcular dónde quedará el punto marcado utilizando sus correspondientes tres puntos en cada fotograma (W_i, Y_i, Z_i) para formar su Bi (como en A_i) y sus lambdas (Λ_i) con la siguiente fórmula:

$$v_i = B_i * \Lambda_i + z_i$$

Ahora sí se pueden comparar los resultados obtenidos del seguimiento con el nuevo *GroundTruth*. Para hallar el error, se va a calcular para cada fotograma la Distancia Euclídea entre los pares de puntos (uno obtenido por el algoritmo y otro el del *GroundTruth*) y después se computa una media por cada punto y por cada fotograma. Después, también se evalúa la robustez de cada punto; para ello se calcula la desviación típica de este error para cada punto a lo largo de todo el vídeo, dividida entre la media calculada.

Estos dos datos se dibujan en una gráfica para que sean más claros. Un ejemplo se puede ver en la **figura 46.** En la gráfica se puede observar en la primera línea los errores en píxeles de los 54 puntos tanto para el seguimiento como para su robustez, en la segunda línea los de los 12 puntos y en la tercera una gráfica del error en píxeles por fotograma de la suma de todos los puntos.

Con esto se evalúa el error de seguimiento.

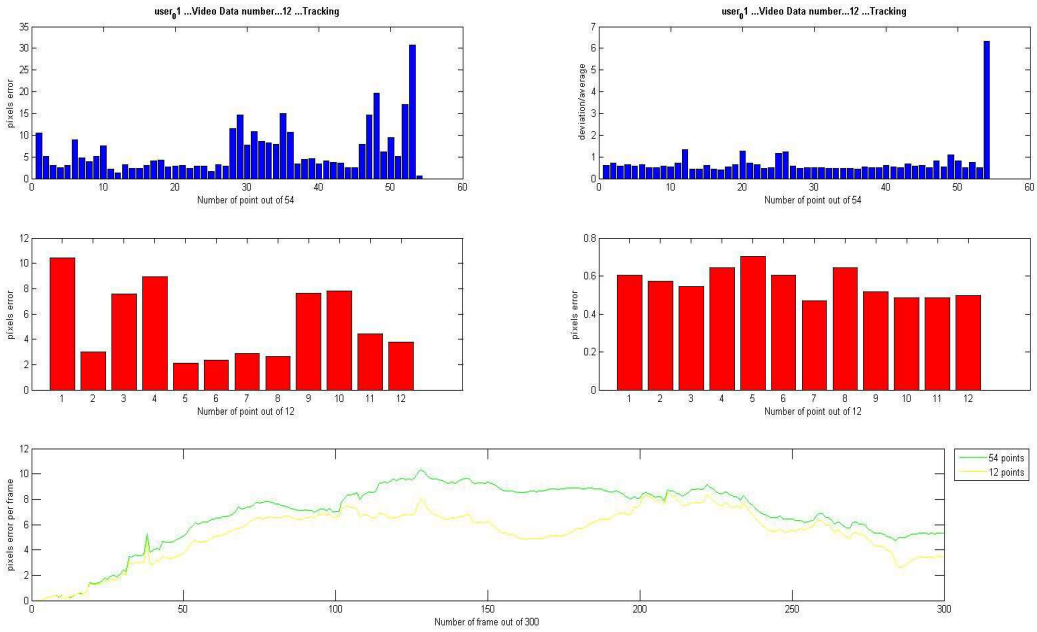


Figura 46: Gráfica de los errores en píxeles por punto y por fotograma para 54 y 12 puntos

3.5.2.2. POSIT

El POSIT es un algoritmo de estimación de la posición y la rotación en tres dimensiones de un objeto respecto a la cámara. Para calcularlo hay que implementar una serie de programas que lo automatizan. El programa principal es el *modernPosit.m*, y requiere de varios datos de entrada. Hasta ahora, ya se ha explicado qué distancia focal y qué centro óptico se toman para cada base de datos de vídeos, pero se necesita saber los puntos en tres dimensiones correspondientes a los puntos de imagen hallados durante el seguimiento. Estos puntos 3D se calculan con un modelo de cabeza genérico en tres dimensiones, el Basel Face Model (BFM)[18] (**figura 47**).

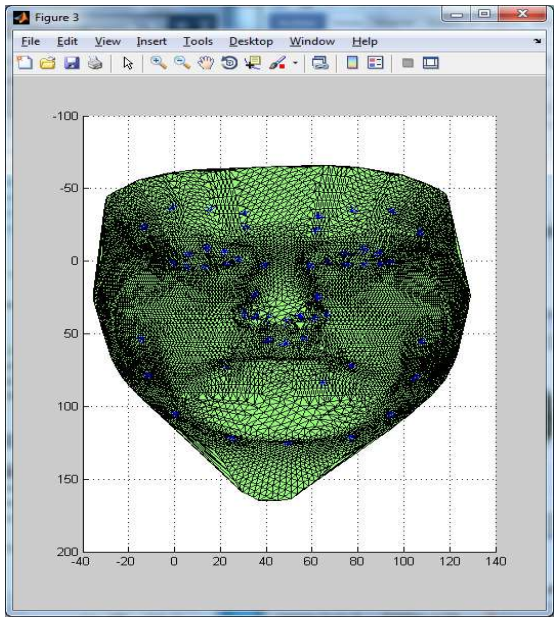


Figura 47: modelo BFM en 3D

Este modelo es la cara promedio obtenida a partir de los escáneres 3D de 200 usuarios, por lo que es razonable utilizarlo como modelo para la estimación de la posición de la cabeza.

El *modernPosit.m* realiza el resto del trabajo. Este programa calcula como salida la traslación y rotación del origen de coordenadas del modelo con respecto a la cámara. Si se le insertan los puntos del modelo en 3D y le indicamos también los puntos correspondientes en 2D que se obtienen en cada fotograma, el *modernPosit.m* devuelve la rotación y traslación calculada para el conjunto global de ellos.

Una vez se tienen las matrices de rotación y traslación, se utiliza otro programa que descompone la matriz de rotación en ángulos de Euler: Roll, Yaw y Pitch.

Tanto en el caso de los vídeos de la UPNA como en los de la BU se dispone, calculado a partir del sensor magnético, del *GroundTruth* de la orientación de la cabeza. Este *GroundTruth* se compara con la estimación obtenida a partir del seguimiento y el POSIT. Un ejemplo se puede observar en las siguientes gráficas, en la primera todos los ángulos detectados en función del número de puntos testados (54, 12, 10, 8, *GroundTruthTracking* y *GroundTruthPOSIT*) y en la segunda la diferencia entre los obtenidos con el Seguimiento y el *GroundTruthPOSIT* (**Figura 48**).

Antes de observar la figura, se debe mencionar un cálculo extra que se ha realizado orientativo para con los demás. Al tener el *GroundTruth* del seguimiento, es interesante ver qué ocurre cuando lo insertamos también en el *modernPosit.m*. Así, utilizando el seguimiento ideal, el error observado corresponde únicamente al ajuste al modelo 3D y al propio error intrínseco de cálculo de POSIT. En la **figura 48** estos resultados también aparecen, siendo denominados *TrackingGTruth*.

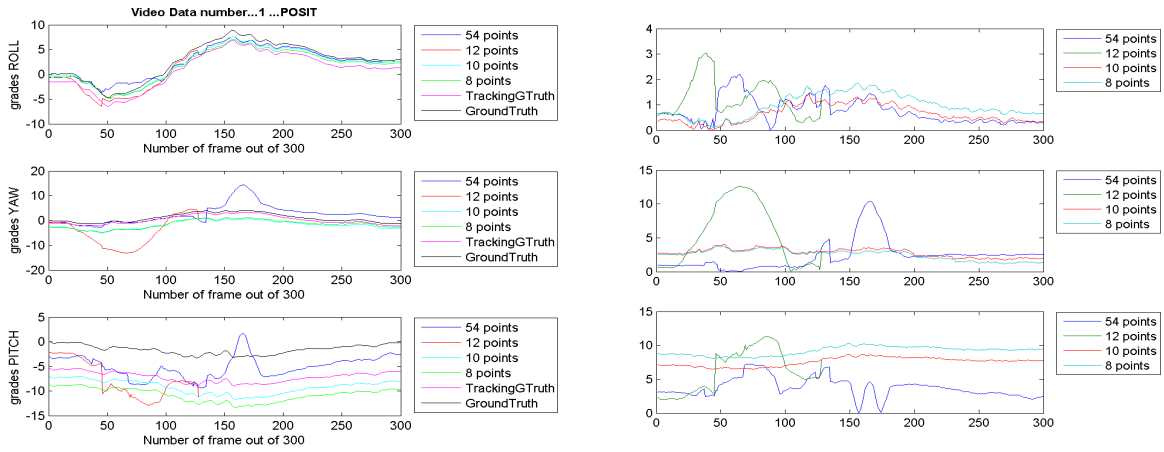


Figura 48: A la izda. gráfica de los ángulos deducidos en Roll, Yaw y Pitch y a la derecha los errores con respecto al GroundTruth

Con esto ya se ha expuesto de manera general el procedimiento de tomar los ángulos del POSIT. El resto del programa evalúa todos los errores obtenidos en seguimiento y en POSIT de cada vídeo dentro de los 120 vídeos en el caso de la UPNA y 45 en la de BU y guarda la media de error de cada uno. Con esto se obtendrán datos numéricos sobre los resultados.

No obstante, falta una última anotación por comentar. En algunos casos, el programa utilizado para ejecutar el POSIT se queda bloqueado o no tiene suficiente información. Este programa funciona únicamente si tiene al menos cuatro puntos no coplanares. Si no tiene estos datos, el programa no obtiene buenos resultados. Por ello, se implementa en el programa una manera de averiguar cuántos fotogramas de cada vídeo se dan por buenos y se obtiene también una medida del éxito de 0 a 1. Si el éxito se considera malo (menor que 0.98) en el promedio de vídeos de todos los usuarios la medida se descarta y no se da por válida. Esto es porque a menor éxito, más posibilidad de que el error sea más bajo (los puntos se van degradando conforme transcurre el tiempo del vídeo).

3.5.3. Implementación de diferentes métodos de Seguimiento

En el apartado anterior se ha explicado el funcionamiento en líneas generales del tratado de errores una vez obtenidos los puntos. Esta sección, en cambio, va a tratar de los diferentes métodos utilizados para obtener los puntos del seguimiento. Se han implementado diferentes maneras, de tal manera que cada una tiene una particularidad diferente a todas las demás. El objetivo es calcular resultados con diferentes algoritmos y después evaluar cuál es el más efectivo, es decir, cuál da el mejor resultado.

Los algoritmos implementados son: Lucas-Kanade sin ningún añadido y utilizando programas ya implementados en Matlab, Lucas-Kanade con validación de puntos, varios tipos de Lucas-Kanade con corrección de descriptores, ASM, AAM e IF y combinaciones de estos tres últimos con Lucas-Kanade.

3.5.3.1. Lucas-Kanade

Este es quizá el algoritmo más sencillo de implementar, porque está prácticamente desarrollado en Matlab. Primero, se va a explicar la idea que hay detrás de él.

El primer paso es inicializar el *Tracker* con los puntos iniciales (marcados anteriormente). Después se evalúa todo el vídeo, utilizando en cada fotograma el

Tracker y obteniendo nuevos puntos para cada escena. Esos nuevos puntos de cada escena se utilizan en la siguiente llamada al *Tracker* como los puntos a seguir.

El algoritmo implementado en Matlab devuelve dos datos con este método, los nuevos puntos y su validez. Así, si un punto es considerado demasiado lejano o impreciso a lo que debería ser, devuelve una validez de 0 y se pierde. Esto es importante tener en cuenta en el tratado de errores, porque por defecto el algoritmo sigue devolviendo valores para esos puntos, que son los valores de la última posición que recuerda haber hallado.

- Optimización en el caso de la base de vídeos de BU

Cuando se obtuvieron los primeros resultados del algoritmo de Lucas-Kanade en la base de vídeos de BU, fueron muy insatisfactorios, dando lugar a errores con el POSIT más grandes que en otras implementaciones en OpenCV.

Al comparar con la base de vídeos de la UPNA, que proporcionaba errores parecidos a los obtenidos con la implementación de las OpenCV, se examinó el algoritmo de Lucas-Kanade y sus parámetros, entre los cuales se puede modificar el tamaño de ventana sobre el que se aplica. Considerando que el valor que estaba por defecto, 30x30 píxeles no era adecuado para la resolución de base de Boston, se procedió a una optimización.

Los resultados de ésta fueron que el mejor valor de tamaño de ventana era 13x13 píxeles, la cual mejoraba los resultados enormemente y los hacía equiparables a las OpenCVs.

Se obtienen los siguientes resultados para la implementación de Lucas-Kanade.

BU:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [13 13]

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	1,87	5,63	**3,62**	1,00	**3,71**
10	**1,86**	5,19	4,17	1,00	**3,74**
8	1,95	**5,12**	4,26	0,99	**3,77**
Mejor	**1,86**	**5,12**	**3,62**	**-**	**3,53**

UPNA:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [31 31] (default)

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,30	5,06	5,46	4,20	1,00	3,94
12	1,33	2,92	3,34	3,28	1,00	2,53
10	**0,77**	**2,83**	**3,02**	-	0,99	**2,20**
8	0,95	2,95	3,14	-	0.99	2,35
Mejor	0,77	2,83	3,02	-	-	**2,20**

3.5.3.2. Lucas-Kanade con validación de puntos

En este algoritmo se empleará una base del algoritmo anterior, ya que se van a utilizar las mismas partes del seguimiento de Lucas-Kanade pero con un añadido. Además de la validación de puntos que ofrece el algoritmo, se va a implementar una validación de puntos extra.

Esta validación se hará utilizando tres posibles variantes. En todas ellas se obtienen los puntos del seguimiento con Lucas-Kanade, y después se realiza un estudio de la ventana de píxeles alrededor de cada uno de ellos. Las ventanas aplicadas son normalmente de 11x11 píxeles, y lo primero que se hace es hallar esa ventana alrededor de cada punto nuevo calculado. Después, se halla la misma ventana alrededor de los puntos iniciales en la escena inicial (primer fotograma) y con estas dos matrices de igual tamaño se realizan diferentes operaciones.

El primer tipo de operación es calcular un valor equivalente a la suma de los cuadrados de las restas de las ventanas, la actual con la inicial. Después, esta suma se

divide entre la desviación estándar de la escena inicial, para establecer la diferencia que hay entre ellas. Este valor se denomina "$\delta_{ActualInit}$" y sigue ésta fórmula:

$$\delta_{ActualInit} = \frac{\sqrt{\sum_n (a_n - P_n)^2}}{std(a)}$$

donde a = ventana de la escena actual,
P = ventana de la escena inicial,
Std = desviación estándar

El segundo tipo de operación es calcular otro valor delta, esta vez también se obtienen ambas ventanas, pero se calcula la correlación entre la resta de ellas. Este valor se denomina "$\delta_{ActualInitCorr}$".

Por último, el tercer tipo de operación es cómo el primero pero sin dividir por la desviación estándar en el caso de la delta Actual-Inicial.

Una vez se han averiguado alguno de estos valores (según la variación que se quiera utilizar) se compararán con dos valores umbrales. El primer valor nos servirá como límite para valores altos (α) y el segundo para valores muy bajos (β), y a partir de aquí se validará el punto según la región.

Si la delta es mayor que α, este punto automáticamente quedará descartado, pues se considera que es muy diferente a lo que se supone bueno en cuanto a calidad. Si la delta es menor que β, este punto automáticamente quedará validado con un 1.

El tercer caso, equivalente a la tercera región y que quedaría entre ambos límites, impone obtener otro tipo de cálculos. Esta vez se obtendrá una delta, tanto en correlación como en diferencias, pero en este caso con la escena previa. Estas deltas obtenidas se denominan "$\delta_{ActualPrev}$" y "$\delta_{ActualPrevCorr}$".

Una vez tenemos estas deltas, se vuelven a comparar con un tercer umbral (ρ), de tal manera que si lo superan se descarta el punto, y si no lo superan la validez se mantiene en 1 y se sigue realizando el seguimiento del mismo.

En el esquema de la **figura 49** se va a resumir el procedimiento para la $\delta_{ActualInit}$, y es análogo para las demás variaciones.

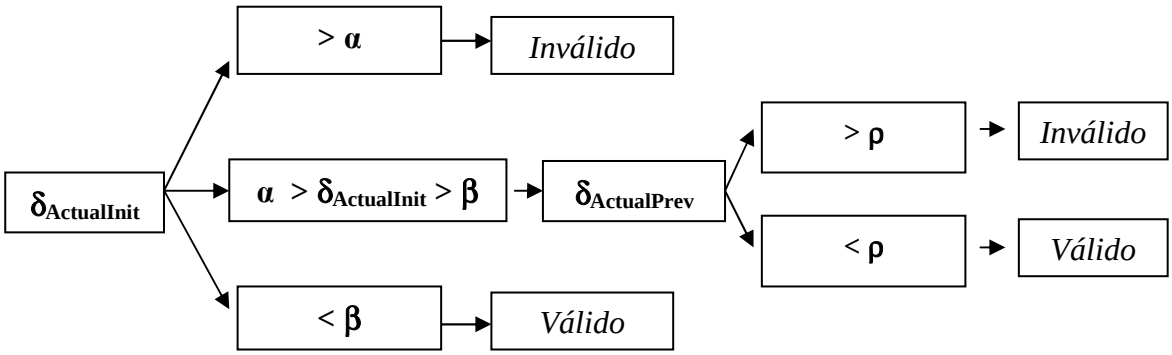


Figura 49: Esquema de funcionamiento método de validación de puntos

Este es el método que se sigue en un principio, aunque existen dos preguntas importantes: ¿cuáles son los umbrales que ofrecen mejores resultados? Y, ¿mejoran los resultados con respecto al algoritmo de seguimiento normal? La segunda cuestión se expondrá mejor en el apartado de resultados, pero para la primera pregunta se van a estudiar los pasos tomados para la resolución de este conflicto.

Una vez implementado el método, se pueden hallar los valores de todas las deltas con Lucas-Kanade con los siguientes valores de umbrales: si α es infinito, β es 0 y ρ es infinito, siempre se calcularán ambas deltas (Inicial-Actual y Previa-Actual) y nunca se invalidará ningún punto.

Una vez realizado esto, se procede a un estudio general de qué valores son los límites entre los que varían las deltas. De manera genérica y para los 12 puntos más característicos, se dibuja los resultados en unas gráficas tipo la que se observa en la **figura 50.**

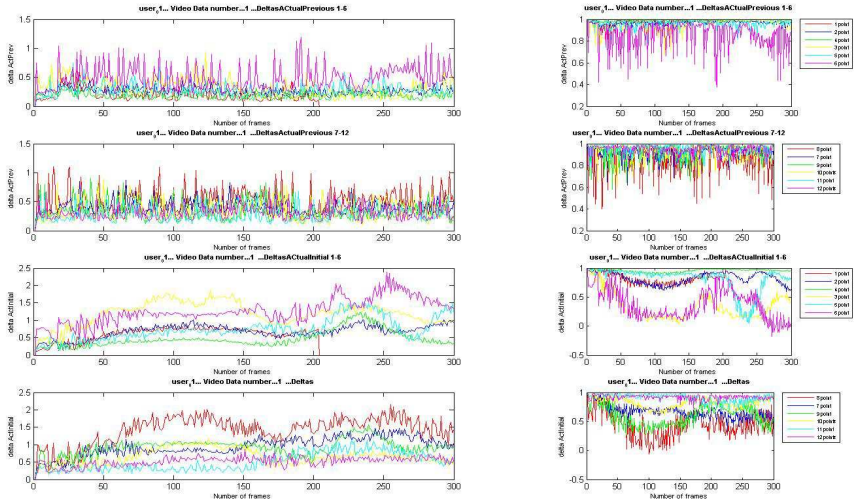


Figura 50: deltas obtenidas de 12 puntos en un vídeo con desviación estándar (izda.)

Optimización del método

En el método de validación de puntos explicado anteriormente subsistía una pregunta sin resolver. ¿Qué valores de los umbrales mejoran el procedimiento? Observando las gráficas se puede ver que para las deltas halladas con todos los métodos los valores en general se encuentran entre varios límites bastante marcados. Interesa averiguar cuál podría ser el valor de umbral para mejorar el método de Lucas-Kanade. También interesa averiguar si las deltas halladas por correlación son mejores o no, o si es mejor aplicar la división entre la desviación estándar o es mejor obviarla.

Para ello, se debe implementar un método de optimización. Matlab permite implementar optimizaciones de manera sencilla, con sólo crear una función en la que se introducen como entrada los valores que se quieren probar y cómo salida la variable que se quiere minimizar. El siguiente paso es, entonces, crear una función, tanto con la base de la UPNA como con la de BU, y a partir de ahí ver qué resultados son los mejores para poner como umbral.

Para ver un ejemplo del algoritmo, se va a escribir cómo queda para la Universidad de Boston. Primero se crea la función:

TotalError = BostonUniversityTesterFunction(ThresholdIN)

Esta función utiliza el algoritmo implementado para obtener resultados con los 45 vídeos, calcula el error total de Yaw, Pitch y Roll de cada vídeo y lo promedia con todos ellos. Este error es la salida "*TotalError*" que se quiere minimizar. Como entrada, se probaran diferentes valores para los tres umbrales en una estructura llamada ThresholdIN. El programa que ejecuta esta optimización es así:

```
optim_init_value = [23.1337 11.2342 3.7071];
lb_value       = [8 5 1];
ub_value       = [30 20 15];
f = @(x)BostonUniversityTesterFunction(x);
options = optimoptions(@fmincon,'Algorithm','interior-point');
problem =
createOptimProblem('fmincon','x0',optim_init_value,'objective',f,'lb',lb_value,'ub',ub_val
ue,'options',options);
gs = GlobalSearch;
evx = run(gs,problem);
```

Y con esto se probaran diferentes valores, en torno a *lb_value* y *ub_value*, y se obtendrán los umbrales que optimizan el método.

BU:

Parámetros:

- Window Size = 10 para todos.
- Umbrales Correlación:

0.3316	0.2075	0.3228

- Umbrales Val con STD:

1,72	0,89	1,89

- Umbrales Val sin STD en la delta Actual-Inicial:

44,2575	39,01	2,6705

Corr

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	1,90	5,10	**3,66**	1,00	**3,55**
10	**1,81**	**4,69**	4,18	1,00	3,56
8	1,92	4,78	4,66	0,99	3,79
Mejor	1,81	4,69	3,66	-	**3,39**

STD

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	**1,87**	4,65	**3,78**	0,98	**3,44**
10	1,89	**4,43**	4,27	0,98	3,53
8	1,94	4,80	4,90	0,98	3,88
Mejor	1,87	4,43	3,78	-	**3,36**

sin STD

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	1,87	5,19	**3,57**	0,99	**3,55**
10	**1,86**	**4,81**	4,14	0,99	3,61
8	1,94	5,10	4,23	0,99	3,75
Mejor	1,86	4,81	3,57	-	**3,42**

UPNA:

Parámetros:

- Window Size = 10 para todos.
- Umbrales Correlación:

0,0030	0,0100	0,0500

- Umbrales Val con STD:

4,0401	18,8800	2,0154

- Umbrales Val sin STD en la delta actual-inicial:

56,8744	39,7408	2,0005

corr

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,28	4,76	5,41	4,02	1,00	3,82
12	1,59	**2,91**	3,79	3,17	0,99	2,76
10	**1,27**	2,97	**3,61**	-	0,99	**2,62**
8	1,40	3,04	3,69	-	0.99	2,71
Mejor	1,27	2,91	3,61	-	-	**2,60**

STD

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,16	4,57	4,49	3,70	1,00	3,41
12	1,09	**2,57**	3,11	2,85	0,99	2,26
10	**0,70**	2,60	**2,85**	-	0,99	**2,05**
8	0,84	2,78	3,10	-	0.98	2,24
Mejor	0,70	2,57	2,85	-	-	**2,04**

sin STD

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,24	4,95	4,47	3,92	1,00	3,55
12	1,28	2,95	3,29	3,13	1,00	2,51
10	**0,78**	**2,88**	**3,03**	-	0,99	**2,23**
8	0,97	3,07	3,18	-	0.99	2,41
Mejor	0,78	2,88	3,03	-	-	**2,23**

3.5.3.3. Lucas-Kanade con corrección de descriptores

Este método está implementado apoyándose también en el seguimiento con el algoritmo de Lucas-Kanade, pero añadiéndole una corrección basada en la utilización de descriptores. Como se ha explicado anteriormente, un descriptor describe un punto característico de la escena con una serie de valores o características especiales calculadas según la región en la que se localiza. Cada descriptor es especial básicamente por la zona en la que está, que hace que determinados valores sean muy específicos y marcados.

Este método comienza utilizando el algoritmo de Lucas-Kanade, y después se realiza una serie de correcciones sobre los puntos obtenidos con este procedimiento. Primero se obtiene una ventana alrededor de cada uno de los puntos de salida, se utilizan tamaños aproximadamente de 5x5 píxeles. Sobre esta ventana de puntos se aplica uno de los algoritmos para obtener los descriptores de cada uno. Estos algoritmos pueden estar basados en el FREAK, el BLOCK, o el SURF.

Así se obtienen los descriptores de todos los puntos de la ventana, y utilizando una función de Matlab de comparación de valores se comparan las características obtenidas en el fotograma anterior de cada punto de Lucas-Kanade con las de la ventana del mismo. Después se toma la mejor comparación como punto resultado y se introduce como punto a seguir para Lucas-Kanade.

En la **figura 51** se resume la forma de trabajar con los descriptores para cada fotograma.

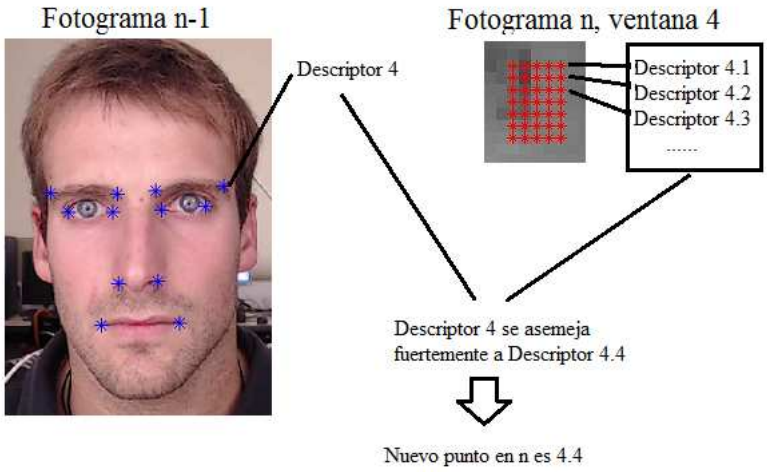


Figura 51: Resumen funcionamiento método con descriptores

Éste es el método general, pero se ha realizado un estudio de este algoritmo separándolo en otros dos dependiendo de un parámetro muy simple.

En el método 3 (se denominará así en un futuro), se comparan los descriptores del fotograma n-1 con respecto a los descriptores del fotograma n. Esto significa que una vez obtenidos los puntos en el fotograma n, se calculan sus respectivos descriptores para ser utilizados en la siguiente iteración.

En cambio, en el método 4, siempre se comparan los descriptores del fotograma n con respecto a los iniciales, los obtenidos en el primer fotograma. Dependiendo de esto los resultados varían, como se verá en el apartado de resultados.

Lucas-Kanade combinado con descriptores anteriores

BU:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [13 13]
- Block
  - MatchValue = 3
  - WindowSize = 5
- Freak
  - MatchValue = 3
  - WindowSize = 5
- Surf
  - MatchValue = 3
  - WindowSize = 5

BLOCK

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	6,56	11,65	**12,74**	1,00	10,32
10	7,03	**10,76**	13,92	1,00	10,57
8	**5,49**	10,97	12,94	0,99	**9,80**
Mejor	5,49	10,76	12,74	-	**9,67**

SURF

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	**3,95**	**12,50**	**9,05**	1,00	**8,50**
10	4,24	12,73	9,89	1,00	8,96
8	4,06	13,35	10,95	0,99	9,45
Mejor	3,95	12,50	9,05	-	**8,50**

FREAK

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	**3,35**	9,22	**13,20**	1,00	**8,5**
10	3,63	**9,21**	14,11	1,00	8,96
8	3,79	9,35	14,58	0,99	9,45
Mejor	3,35	9,21	13,20	-	8,5

UPNA:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [31 31] (default)
- Block
  - MatchValue = 3
  - WindowSize = 5
- Freak
  - MatchValue = 3
  - WindowSize = 5
- Surf
  - MatchValue = 3
  - WindowSize = 5

BLOCK

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	**2,46**	5,13	**7,81**	10,80	1,00	**5,14**
12	3,24	**4,97**	7,86	10,61	0,99	5,35
10	3,07	6,11	8,60	-	0,99	5,93
8	3,18	6,32	8,55	-	0.99	6,02
Mejor	2,46	4,97	7,81	-	-	**5,08**

SURF

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	5,08	5,57	8,32	1,00	2,49	6,32
12	5,10	5,46	8,32	1,00	2,47	6,29
10	5,11	5,37	8,30	-	0,99	6,26
8	**5,05**	**5,22**	**8,26**	-	0.99	**6,17**
Mejor	5,05	5,22	8,26	-	-	**6,17**

FREAK

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	**1,74**	4,76	5,94	8,03	1,00	4,15
12	2,16	**4,30**	**5,49**	6,60	0,99	**3,98**
10	2,36	4,87	5,62	-	0,99	4,28
8	2,60	4,94	5,67	-	0.99	4,40
Mejor	1,74	4,30	5,49	-	-	**3,84**

Lucas-Kanade combinado con descriptores iniciales

BU:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [13 13]
- Block
  - MatchValue = 3
  - WindowSize = 5
- Freak
  - MatchValue = 3
  - WindowSize = 5
- Surf
  - MatchValue = 3
  - WindowSize = 5

BLOCK

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	2,37	6,32	5,33	1,00	4,67
10	2,10	5,34	5,96	1,00	4,47
8	**2,04**	**5,29**	**5,24**	0,99	**4,19**
Mejor	2,04	5,29	5,24	-	**4,19**

FREAK

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	2,01	5,72	**4,28**	0,99	4,00
10	**1,97**	**5,06**	4,67	0,99	**3,90**
8	2,03	5,48	4,86	0,99	4,12
Mejor	1,97	5,06	4,28	-	**3,77**

SURF

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	2,52	6,21	5,13	1,00	4,62
10	2,66	**5,75**	5,21	1,00	4,54
8	**2,28**	6,15	**5,11**	1,00	**4,51**
Mejor	2,28	5,75	5,11	-	**4,38**

UPNA:

Parámetros:

- Initialize Tracker
  - MaxBidirectionalError = 1
  - BlockSize = [31 31] (default)
- Block
  - MatchValue = 3
  - WindowSize = 5
- Freak
  - MatchValue = 3
  - WindowSize = 5
- Surf
  - MatchValue = 3
  - WindowSize = 5

BLOCK

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54,00	1,68	4,57	6,24	5,33	1,00	4,16
12,00	1,81	4,00	4,88	4,50	0,99	3,56
10,00	1,33	3,84	5,38	-	0,99	3,52
8,00	**1,17**	**3,56**	**4,61**	-	0.99	**3,11**
Mejor	1,17	3,56	4,61	-	-	**3,11**

FREAK

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,47	5,07	5,35	4,75	1,00	3,96
12	1,49	2,93	3,41	3,17	1,00	2,61
10	**0,89**	**2,54**	**3,26**	-	0,99	**2,23**
8	1,01	2,63	3,34	-	0.99	2,32
Mejor	0,89	2,54	3,26	-	-	**2,23**

SURF

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,45	5,22	5,85	4,80	1,00	4,17
12	1,31	3,13	**3,28**	3,02	0,99	2,57
10	**0,84**	**2,91**	3,29	-	0,99	**2,34**
8	0,88	2,97	3,44	-	0.99	2,43
Mejor	0,84	2,91	3,28	-	-	**2,34**

3.5.3.4. SIFT

Este método tiene poco que ver con los demás, y se implementa de manera muy diferente, como se verá más adelante. Como resumen, se trata de obtener los descriptores de los puntos más característicos de cada fotograma por el método SIFT de y averiguar las correspondencias o "*matchings*" entre cada pareja *n-1* y *n* de ellos, quedarnos con aquellos puntos que nos interesan dentro de la cara del sujeto en la imagen y a partir de ahí estimar la posición y rotación de la cabeza.

Debido a esto, se tiene que partir de la base de que el SIFT y sus *matchings* sean suficientemente buenos y precisos, sino el resultado del algoritmo será problemático y no se obtendrán buenos resultados. Así, se deriva que el primer paso es una evaluación

del propio método SIFT con respecto del seguimiento, y después implementar el algoritmo que realiza POSIT y los resultados finales.

Seguimiento en el método SIFT:

En este apartado se va a evaluar el seguimiento de la base de datos de la UPNA con la función de SIFT implementada por Artiom Kovnatsky[14]. Para ello, lo que se hace primero es obtener las parejas de puntos característicos que detecta esta función en cada fotograma y su siguiente, de tal manera que así obtenemos los correspondientes puntos de interés entre las parejas de fotogramas. Al ser el coste temporal de la función muy alto (alrededor de cinco horas por vídeo), se obtienen los datos de cada vídeo número 7 de cada usuario, utilizando en total 10 vídeos en los que hay un movimiento libre de la cabeza y con usuarios diferentes para trabajar.

Una vez tenemos las parejas de puntos relevantes en cada pareja de fotogramas, podemos empezar a evaluar el Seguimiento. Así, se carga el vídeo que se va a calcular, se cargan los 12 puntos del primer fotograma característicos y se implementa el resto del método con el SIFT. Este método sigue los pasos escritos en una publicación[21] sobre la evaluación de este método.

Los pasos a seguir son los siguientes. En cada fotograma se calcula la pose y a partir de ella se siguen calculando los datos. No obstante, para ver si este método funciona con el seguimiento, la pose la vamos a tomar del *GroundTruth*, de tal manera que siempre tenemos un valor preciso y podemos ver cómo actúa la función SIFT obtenida de Internet únicamente respecto al seguimiento de los puntos.

El resto del método está explicado en el siguiente apartado, así que aquí se van a poner los resultados del seguimiento, con un POSIT ideal, a partir de la función del SIFT descargada y se comparará con el error de seguimiento del método de Lucas-Kanade para los 12 puntos más característicos.

Existe un detalle que se debe mencionar, antes de ver los resultados, y es que las relaciones entre los puntos sacadas con la función SIFT tal cual eran bastante irregulares, teniendo algunas correspondencias muy precisas y otras que se separaban más de 40 píxeles de un fotograma al siguiente, lo cual en teoría no puede ocurrir, ya que los movimientos de la cabeza no son tan rápidos como para desplazarse tanto en tan poco tiempo. Por ello, se hace una modificación en los puntos obtenidos con el SIFT de tal manera que si la Distancia Euclídea entre el punto en el fotograma *n-1* y el mismo punto en el fotograma *n* supera un valor, esa correspondencia se elimina.

En base a éste cálculo de la Distancia Euclídea, se han realizado dos aproximaciones. Una calculando que la Distancia tenga que ser menor de 5 píxeles (S.

15), que en general los resultados son peores pero abarcan más vídeos, y otra que sea menor que 5 píxeles (S. 5).

S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK
1	**1**	**1**	**2**	**2**	**2**	**3**	**3**	**3**	**4**	**4**	**4**	**5**	**5**	**5**
3,2	3,1	8,5	1,0	1,0	31,8	0,4	0,4	5,5	4,1	4,1	7,0	-	-	2,3
16,4	11,8	2,8	3,9	4,1	4,7	3,6	4,2	2,6	7,2	10,2	7,3	9,6	7,9	2,4
8,2	7,7	3,7	9,5	5,4	3,7	10,7	16,1	4,8	8,6	7,2	4,9	5,7	2,7	3,5
1,9	1,9	6,6	1,2	0,7	8,6	5,4	8,1	6,6	1,5	1,4	4,5	1,1	0,6	4,6
5,7	8,8	5,3	9,8	9,6	21,6	6,7	7,3	2,9	9,1	6,0	0,4	12,2	3,3	2,7
14,3	12,4	0,2	4,5	4,9	3,5	9,5	9,0	2,9	7,1	6,6	0,3	9,5	17,8	2,2
11,7	6,2	0,2	12,1	3,8	2,9	19,8	11,7	1,8	6,1	3,7	0,3	14,1	6,4	3,3
4,7	3,3	0,2	4,2	2,3	8,0	1,9	1,7	3,5	15,9	11,9	0,3	12,1	4,0	3,8
7,7	9,9	3,2	3,4	3,5	3,8	8,2	5,3	2,4	3,3	3,3	2,6	5,1	7,7	2,1
6,5	6,7	3,4	9,4	8,6	2,5	8,3	4,6	2,1	5,2	3,6	2,6	9,4	14,0	1,8
11,5	9,4	3,0	2,4	2,2	0,8	8,7	8,1	1,4	24,1	9,3	2,5	2,7	2,0	2,0
11,0	6,4	3,2	2,4	2,4	3,0	9,4	9,1	1,9	9,7	9,6	2,3	17,4	13,0	2,5
8,6	**7,3**	**3,4**	**5,3**	**4,0**	**7,9**	**7,7**	**7,1**	**3,2**	**8,5**	**6,4**	**2,9**	**9,0**	**7,2**	**2,8**

S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK	S. 15	S. 5	LK
6	**6**	**6**	**7**	**7**	**7**	**8**	**8**	**8**	**9**	**9**	**9**	**10**	**10**	**10**
3,5	-	9,3	1,0	-	5,7	0,2	0,2	7,0	-	-	6,9	0,0	0,0	4,6
10,5	-	4,4	7,4	-	4,9	5,5	4,7	3,4	9,4	-	3,8	6,8	5,0	3,0
6,5	-	6,8	9,6	-	5,3	8,5	4,3	3,4	14,3	-	5,6	14,1	18,1	2,8
2,6	-	10,0	1,2	-	5,6	10,7	4,0	3,6	5,5	-	6,5	14,1	18,1	3,6
5,6	-	7,0	2,0	-	4,4	5,4	3,2	3,1	1,7	-	0,0	4,1	4,0	2,3
10,1	-	3,4	11,0	-	5,3	5,0	3,9	3,4	5,5	-	3,5	6,0	11,2	2,7
7,6	-	7,3	16,7	-	4,3	16,2	5,0	3,3	4,6	-	3,5	22,5	14,4	2,6
3,0	-	1,5	8,3	-	4,7	8,2	2,1	3,3	16,5	-	0,0	3,3	2,4	2,9
7,0	-	5,3	9,9	-	4,4	4,0	4,3	2,8	11,1	-	2,6	4,3	9,8	2,7
6,1	-	3,8	7,0	-	4,5	8,8	2,5	3,0	2,4	-	2,4	4,2	12,4	3,0
5,8	-	5,1	5,8	-	4,7	3,6	2,7	2,6	1,0	-	2,5	9,6	7,0	2,5
7,8	-	4,8	1,3	-	5,8	8,2	19,1	2,4	14,6	-	2,5	6,0	5,2	2,1
6,3	**-**	**5,7**	**6,8**	**-**	**5,0**	**7,0**	**4,7**	**3,4**	**7,9**	**-**	**3,3**	**7,9**	**9,0**	**2,9**

Hay valores en las tablas que no tienen ningún número. Esto es porque en según qué vídeos al eliminar tantas correspondencias no había suficientes puntos como para seguir calculando el seguimiento y se pierde el seguimiento del vídeo.

Aparte de esto, en los datos se observa que sólo en dos vídeos de los 10 los resultados del método SIFT son mejores que Lucas-Kanade, obteniéndose datos realmente malos en los demás, con errores de seguimiento medios por punto de más de 5 y 7 píxeles.

Con estos resultados, se desecha este método de momento, mientras no se posea una función de evaluación del SIFT mucho más precisa.

Implementación del método SIFT:

En el método SIFT no se va a utilizar Lucas-Kanade. De manera resumida, este algoritmo funciona así: primero carga el vídeo para el que se va a utilizar, después toma un fotograma y el siguiente y calcula, a través de la función de SIFT descargada los puntos característicos de cada uno y sus respectivos enlaces entre ellos. A partir de aquí se limita a recoger sólo los puntos dentro la cara y se utiliza el POSIT con esos nuevos puntos.

Sin embargo, este proceso es más complicado de implementar. Se va a explicar paso por paso tal y como se encuentra en el trabajo en el que está basado[21].

El primer punto es la inicialización. Hay varios parámetros que se requieren antes de empezar con el proceso, entre ellos el modelo 3D que se va a utilizar y sus respectivas proyecciones, así como la posición de la cabeza inicial (en el primer fotograma). Como ya se tiene el modelo 3D y los puntos marcados en el primer fotograma, se utiliza el POSIT con *modernPosit.m* y se calcula la estimación de la posición de la cabeza. Después, se calcula la proyección del modelo en la imagen porque luego se utilizará en el siguiente punto. En la **figura 52** se observa una imagen de cómo queda el modelo 3D proyectado.

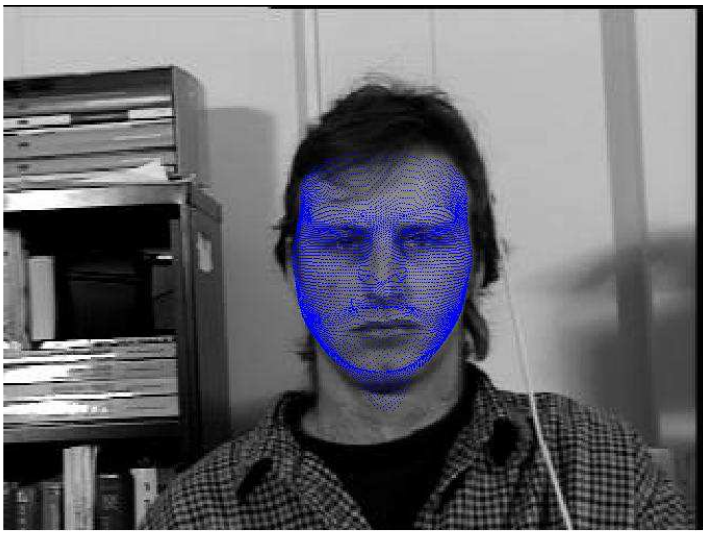


Figura 52: Proyección del modelo BFM sobre el vídeo 1 de la BU

El siguiente paso es, una vez se tiene la proyección y la posición inicial de la cabeza, se comienza a cargar el vídeo. Así se obtiene el fotograma *n*, que será el actual, y el fotograma *n-1*, que será el previo. Con estos dos fotogramas, calculamos con el SIFT los puntos característicos de la imagen y sus respectivas uniones. En la **figura 53** se puede apreciar mejor cómo funciona.

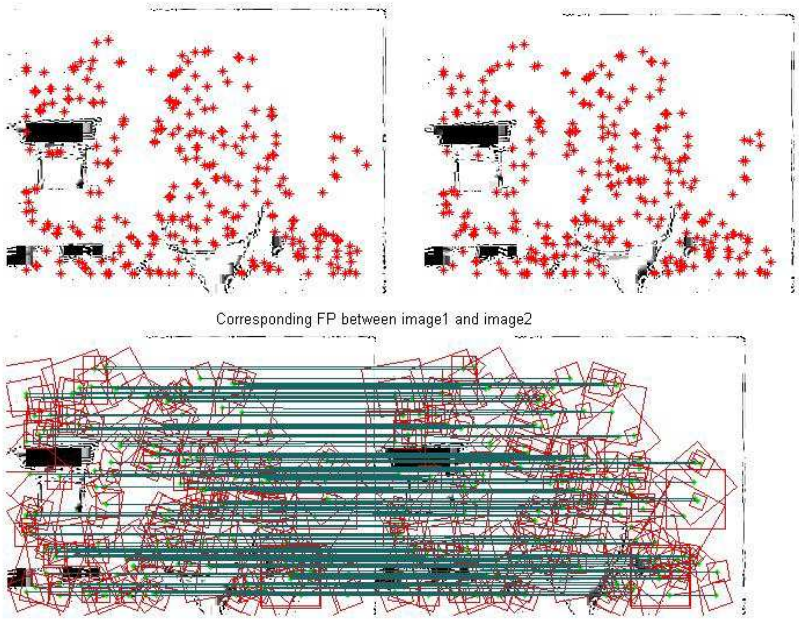


Figura 53: Correspondencias o "matchings" entre dos fotogramas de un vídeo de la BU

Ahora tenemos los puntos de cada fotograma que son más característicos para el SIFT y los tenemos relacionados en dos tablas: "*PrevPoints*" y "*ActualPoints*". Sin embargo, de estos puntos muchos quedan localizados fuera de nuestra zona de interés, la cara, así que tenemos que eliminarlos de alguna manera. Se utiliza la proyección del modelo 3D sobre la imagen para saber dónde están los bordes de la zona que nos interesa.

Así, con esta proyección, se usan las funciones *convhull* e *inpolygon* (ya implementadas en Matlab) para delimitarla y obtener los puntos característicos que caen dentro de la región. En la **figura 54** se fotografía el resultado.

Figura 54: Puntos de correspondencias o "*matchings*" con los que nos quedamos

Una vez se tienen los puntos dentro de la región de la cara, se utilizan para delimitar un nuevo polígono entre ellos y seleccionar qué puntos del modelo 3D son los correspondientes dentro de esta nueva zona. Estos nuevos puntos del modelo 3D son con los que se va a trabajar al final. El proceso es muy similar utilizando las mismas funciones y se puede observar en la **figura 55**.

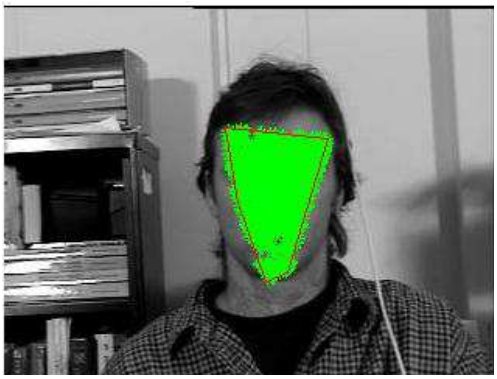


Figura 55: Puntos 3D del modelo que podemos triangular

Los puntos en verde son los puntos del modelo 3D proyectados que van a interesar. El siguiente paso es, a partir de los puntos del SIFT obtenidos en el fotograma *n-1* y los obtenidos en el fotograma *n* se realizan una serie de triangulaciones con coordenadas baricéntricas y se calculan las nuevas coordenadas 2D de los puntos del modelo en el fotograma actual. En la **figura 56** se observan en rojo los puntos correspondientes de todo el modelo 3D proyectado que caerían dentro de la zona delimitada en el fotograma *n*.

Figura 56: puntos 3D obtenidos después de las triangulaciones

Con esto ya se han obtenido los puntos 2D en el fotograma *n* y se tienen sus correspondientes 3D. Ahora se puede calcular el POSIT con ellos. Hay que tener en cuenta que, debido a la gran cantidad de cálculos y al procesado de la imagen, hay bastantes puntos que se denominan *outliers* y que son puntos que no se consideran válidos.

En el documento del que se ha extraído toda la información sobre la implementación de este método, queda poco claro cómo procesan los datos obtenidos para deshacerse de los *outliers*. En esta implementación se decide utilizar todos los puntos calculados como entrada del *modernPosit.m*, y así obtener una medida promediada de la posición de la cabeza..

Como ya hemos explicado en el apartado anterior, en la implementación de SIFT sólo se han utilizado, por tiempos que ocupa de procesado, los vídeos número 7 de todos los usuarios. Después de los resultados del seguimiento, sólo se ha experimentado por ver qué resultados se obtenían. Estos resultados se obtienen corrigiendo la salida de la función SIFT con correspondencias que disten entre sí menos de 15 píxeles.

Parámetros:

- Modelo reducido en [0,91 0,93 1]
- Datos de SIFT corregidos con distancias de menos de 15 píxeles

UPNA:

Usuario	Video nº	ROLL	YAW	PITCH
1	**7**	2,88	2,74	8,33
2	**7**	5,73	8,10	6,09
3	**7**	5,49	7,79	7,00
4	**7**	8,66	9,79	6,13
5	**7**	3,82	7,71	6,18
6	**7**	9,91	9,74	8,45
7	**7**	2,72	3,05	8,04
8	**7**	3,05	3,88	5,10
9	**7**	2,11	2,89	4,12
10	**7**	2,40	9,96	4,34
	Promedio	**4,68**	**6,57**	**6,38**

3.5.3.5. IntraFace

En la implementación del IntraFace se han utilizado puntos ya obtenidos en el Trabajo Fin de Master de José Javier Bengoechea[19] y se han insertado en el programa de evaluación sin modificarlos. Se trata de obtener los resultados de seguimiento y de la posición de la cabeza para cada fotograma de cada vídeo.

En el caso de la evaluación del seguimiento, el IntraFace es un programa que utiliza 49 puntos (**figura 6**), estos puntos difieren con los 54 que se obtienen del útil y que se han estado aplicando hasta ahora. Debido a esto, en el cálculo del nuevo *GroundTruth* por coordenadas baricéntricas que se realiza para los 54 puntos se deben modificar algunos aspectos. Esta diferencia radica únicamente en las relaciones entre los puntos, como ahora son 49 y están localizados en diferentes posiciones, en algunos casos los puntos con los que se triangula del antiguo *GroundTruth* varían.

Por ello se desarrolla otra tabla de puntos de triangulación para convertir el *GroundTruth* de la UPNA y que se pueda utilizar con el IntraFace:

PUNTO	A	B	C		PUNTO	A	B	C
1	1	2	11		**26**	9	6	23
2	1	2	11		**27**	6	10	42
3	2	12	3		**28**	6	9	42
4	3	14	4		**29**	6	9	42
5	4	15	5		**30**	6	9	42
6	9	23	10		**31**	6	9	42
7	8	22	9		**32**	29	47	50
8	7	21	9		**33**	31	49	50
9	7	20	8		**34**	31	33	49
10	7	6	19		**35**	31	33	50
11	27	37	32		**36**	31	33	51
12	27	37	32		**37**	33	50	51
13	27	37	32		**38**	35	43	52
14	27	37	32		**39**	42	44	51
15	27	37	31		**40**	42	44	51
16	27	37	31		**41**	38	42	50
17	27	37	32		**42**	38	42	49
18	27	37	33		**43**	38	42	49
19	27	37	35		**44**	38	42	50
20	1	18	3		**45**	38	42	50
21	3	11	17		**46**	38	42	50
22	3	4	16		**47**	38	42	50
23	5	16	27		**48**	38	42	50
24	2	4	28		**49**	38	42	50
25	1	4	29					

Cabe mencionar que en cuatro puntos concretos, los pertenecientes al tabique nasal, el *GroundTruth* calculado no es correcto. Al ser puntos que se alejan fuertemente de la coplanaridad con el resto, no se acaban de triangular bien y visualmente se observa que se alejan de los puntos reales. Por ello, en estos datos no se puede tener en cuenta el error de fotograma calculado porque no es real. Sin embargo, en los datos guardados se pueden observar los errores del resto de los puntos, siendo bastante pequeños.

Después de obtener los nuevos datos, se introducen los puntos del IntraFace en el procesado del POSIT y se obtienen los nuevos resultados con el modelo utilizado.

BU:

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	**1,73**	3,80	3,47	1,00	3,00
10	1,77	**3,64**	**3,40**	1,00	**2,94**
8	1,79	3,78	3,86	1,00	3,14
Mejor	1,73	3,64	3,40	-	**2,92**

UPNA:

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
49	0,50	2,08	**2,52**	2,70	1,00	**1,70**
12	0,64	**1,89**	4,10	2,45	1,00	2,21
10	**0,42**	2,10	3,53	-	1,00	2,02
8	0,44	1,96	3,51	-	1,00	1,97
Mejor	0,42	1,89	2,52	-	-	**1,61**

3.5.3.6. ASM

Para la realización de los resultados en ASM la implementación ha sido parecida a la del IntraFace. Los puntos del seguimiento han sido proporcionados por el Trabajo de José Javier Bengoechea[19]. Así, lo único que se ha hecho es variar el programa principal levemente para que el seguimiento consista en procesar los puntos obtenidos por José Javier, de tal manera que luego éstos se evalúan igual que los demás. El resto de programa, una vez dispuesto el seguimiento, es el mismo que para Lucas-Kanade.

BU:

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	3,86	**8,42**	7,24	1,00	**6,51**
10	3,86	8,93	7,11	1,00	6,64
8	**3,84**	8,95	**6,80**	1,00	6,53
Mejor	3,84	8,42	6,80	-	**6,35**

UPNA:

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,92	3,53	3,68	3,90	1,00	2,71
12	0,91	**2,78**	3,36	3,50	1,00	**2,35**
10	**0,89**	2,97	3,30	-	1,00	2,39
8	**0,89**	2,97	**3,28**	-	1,00	2,38
Mejor	0,89	2,78	3,28	-	-	**2,32**

3.5.3.7. ASM – LK

Otro de los algoritmos en los que se ha estado interesado en desarrollar es la combinación del método ASM con Lucas-Kanade en la base de datos de la UPNA, viendo los buenos resultados de ambos algoritmos.

Los puntos obtenidos por cada uno de los métodos se pueden utilizar de varias maneras para implementar mejoras en la obtención del HPE.

Reasignación de Lucas-Kanade (*set or not set*)

La primera variante que hay que tener en cuenta es que los puntos de Lucas-Kanade se pueden reasignar cada fotograma, esto es, decirle al algoritmo en cada escena qué puntos tiene que seguir en la siguiente. En este caso los puntos útiles de Lucas-Kanade van cambiando cada vez, así que se podría decir que el algoritmo está siendo utilizado en tiempo real. Por otro lado, se puede utilizar el algoritmo con los puntos marcados sólo al inicio y no reasignarlos durante el vídeo.

La ventaja de cambiar cada fotograma para Lucas-Kanade es que en cada escena los puntos son más precisos, siempre y cuando sean perfectamente correctos. Sin embargo, si esto no se cumple puede ser fuente de errores: si el punto no es exacto, Lucas-Kanade asimila información de otro punto que, aunque sea muy cercano, no es el característico, y esto a la larga da lugar a fallos más visibles.

Corrección de fallos de ASM:

En el desarrollo del algoritmo también hay que tener en cuenta que el modelo ASM obtiene en general resultados estables, pero en algunos fotogramas concretos falla fuertemente, alejándose el modelo completamente de la cara. Esto se puede observar en la **figura 56**.

Para corregir este fallo, se han propuesto dos soluciones y se ha elegido la que parecía tener mejores resultados en vídeos aleatorios.

Se comentan brevemente ambas opciones. La primera manera de descartar los puntos del ASM cuando el modelo está lejos de la imagen es calcular el centro de los puntos válidos de LK y compararlo con el centro de los mismos puntos de ASM. Si la distancia es mayor que un cierto umbral (se elige 15 píxeles), se presupone que los puntos de ASM están fuera de la cara (**figura 57**). El centro calculado por LK se considera correcto porque aunque algunos puntos puedan estar alejados de su

localización y ser inexactos, el algoritmo funciona de manera independiente para cada punto y el resultado global será más cercano al verdadero centro de la nube.

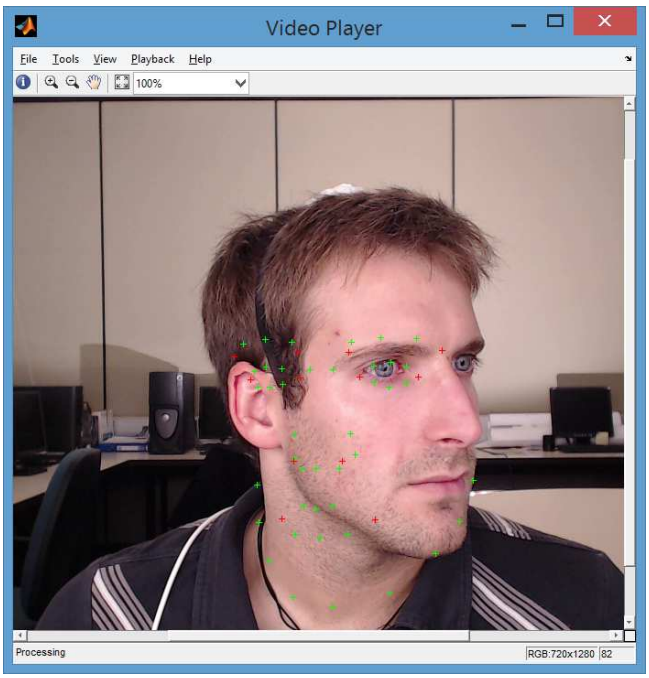


Figura 57: Puntos del ASM en un fotograma dónde no se capturan bien.

La segunda manera es calcular las distancias entre puntos de LK y ASM por separado y tomar el menor valor. Si este menor valor supera un umbral (se prueba con 1 píxel) se descartan los puntos de ASM. De estas dos medidas, se decide tomar la primera como método para corregir este pequeño problema que surge con ASM.

Se elige esta manera porque en varios vídeos elegidos al azar obtiene mejores resultados para todos ellos.

Implementación:

A parte de los cambios antes mencionados que se aplican en todos los casos, también se han probado cuatro maneras de combinar ASM y LK. Para implementar estas variaciones, primero hay que tener en cuenta todas las posibilidades entre los puntos del ASM y Lucas-Kanade. Esto es, ver en que fotogramas son válidos y cuándo no, y actuar en consecuencia. Asimismo hay que utilizar la distancia entre los centros hallada anteriormente para saber si ASM está funcionando correctamente. En función de esto, se implementan varias posibilidades para cada punto testado.

Si todo está en orden y el punto es válido tanto en Lucas-Kanade como en ASM, y la distancia entre los centros de ambas nubes es menor que 15 píxeles, entonces se efectuará una de las cuatro propuestas para combinar los métodos.

No obstante, puede ocurrir que el punto de Lucas-Kanade sea válido y no lo sea el punto obtenido con el método ASM. En este caso, se elegirá como salida el punto de Lucas-Kanade. Se procede de la misma manera si la distancia entre los centros de ambas nubes (ASM y Lucas-Kanade) es mayor que 15 píxeles, ya que esto significa que la

nube de puntos de ASM se ha alejado completamente de la cara en la imagen y por tanto sus puntos están mal localizados.

Otro de los casos que se puede dar es que los puntos de ASM sean válidos y no lo sean los de LK, entonces se utilizarán los puntos de ASM si y sólo si la distancia es menor que 15 píxeles (es decir, si se puede confiar en la nube de ASM).

Por último, si se está ante cualquier otro caso, se invalidará el punto y se pondrá un valor aleatorio, puesto a (100,100) por defecto. En la siguiente tabla se hace un pequeño resumen para aclarar las posibilidades.

Validez LK	**Validez ASM**	**Distancia entre centros**	**Opción**	**Validez**
1	1	< 15	Uno de los 4 métodos	1
1	0	No importa	Puntos LK	1
1	No importa	> 15		1
0	1	< 15	Puntos ASM	1
Otro caso			100,100	0

Se va a explicar cada uno de los cuatro métodos pensados para combinar LK y ASM en caso de que ambos puntos sean válidos.

3.5.3.7.1. Delta inicial-actual

El primero de los métodos que se ha implementado para combinar Lucas-Kanade con ASM consiste en calcular las deltas de cada punto, tal y como se hacía en el método 2 de validación de puntos y combinarlos a partir de ellas. Resumiendo, se calcula una ventana de píxeles alrededor de cada punto en el fotograma actual y se compara con la misma ventana en el fotograma inicial, así se calcula una delta:

$$\delta_{ActualInit} = \frac{\sqrt{\sum_n (a_n - P_n)^2}}{std(a)}$$

Esto se hace con los puntos de Lucas-Kanade y con los puntos de ASM. Después se crea un peso para cada punto con el siguiente cálculo:

$$\omega_{LKinit} = 1 - \frac{\delta_{LKinit}}{\delta_{LKinit} + \delta_{ASMinit}}$$

Este peso es el que se le va a otorgar a los datos obtenidos por Lucas-Kanade para ese punto, y (1-ω_{LK}) es el peso que se le va a dar a los datos del ASM en el mismo, de tal manera que el punto final es una ponderación de la siguiente forma:

$$\text{OutPoint} = \omega_{LKinit} * (x, y)_{LK} + (1 - \omega_{LKinit}) * (x, y)_{ASM}$$

Sin reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,82	4,34	4,58	3,07	1,00	3,25
12	0,66	**2,27**	**2,59**	2,51	1,00	**1,84**
10	**0,53**	2,44	2,63	-	1.00	1,86
8	0,59	2,42	2,60	-	1,00	1,87
Mejor	0,53	2,27	2,59	-	-	**1,79**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,81	3,15	3,39	3,33	1,00	2,45
12	0,79	**2,52**	3,11	2,96	1,00	**2,14**
10	**0,78**	2,73	3,09	-	1,00	2,20
8	**0,78**	2,74	**3,07**	-	1,00	2,20
Mejor	0,78	2,52	3,07			**2,12**

3.5.3.7.2. Delta inicial-actual previa-actual

El segundo método implementado para la mejora es muy parecido al anterior, sólo que calcula también la delta del fotograma actual con el previo y en función de eso se calcula un nuevo peso:

$$\omega_{LKprev} = 1 - \frac{\delta_{LKprev}}{\delta_{LKprev} + \delta_{ASMprev}}$$

$$\omega_{LK} = \frac{\omega_{LKinit} + \omega_{LKprev}}{2}$$

Este peso ω_{LK} es el que se le va a otorgar a los datos obtenidos por Lucas-Kanade para ese punto, y $(1-\omega_{LK})$ es el peso que se le va a dar a los datos del ASM, de tal manera que la localización del punto de salida al final es:

$$\text{OutPoint} = \omega_{LK} * (x, y)_{LK} + (1 - \omega_{LK}) * (x, y)_{ASM}$$

Sin reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,82	4,28	4,53	3,06	1,00	3,21
12	0,67	**2,29**	**2,59**	2,52	1,00	**1,85**
10	**0,54**	2,45	2,62	-	1,00	1,87
8	0,60	2,44	**2,59**	-	1,00	1,88
Mejor	0,54	2,29	2,59	-	-	**1,80**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,80	3,15	3,38	3,32	1,00	2,44
12	0,78	**2,49**	3,11	2,95	1,00	**2,13**
10	**0,77**	2,71	3,10	-	1,00	2,19
8	0,78	2,73	**3,08**	-	1,00	2,20
Mejor	0,77	2,49	3,08	-	-	**2,11**

3.5.3.7.3. Cálculo de rotación en la escena

En este caso, también se va a dar un valor de peso a los datos de ASM y de Lucas-Kanade en función de las deltas inicial y final, pero ahora esas deltas también van a tener otro peso añadido sobre ellas.

Dependiendo de la rotación del sujeto en cada fotograma, la delta Actual-Inicial tiene más fuerza o no. En el fotograma inicial, el sujeto está prácticamente frontal, por ello, si la rotación del sujeto en el fotograma actual es muy grande, la delta que se calcula con la escena inicial no es correcta, porque el sujeto estará en una posición muy diferente.

Para calcular esto, se calcula en cada escena la posición de la cabeza y se analiza la rotación general en Roll, Yaw y Pitch. Una vez se ha calculado, se suma y se divide entre 30º. Se elige 30º como umbral de rotación máxima. Después, una vez calculado este nuevo peso, se evalúa como sigue:

$$\omega_{LK} = \frac{(1 - RotDeg) * \omega_{LKinit} + RotDeg * \omega_{LKprev}}{2}$$

$$\text{OutPoint} = \omega_{LK} * (x, y)_{LK} + (1 - \omega_{LK}) * (x, y)_{ASM}$$

donde RotDeg es el valor calculado de la rotación. Cómo se puede observar, si la rotación es 0º, se le da máximo peso a la ω_{LK} inicial, puesto que se tratará de un objeto colocado en perfecta frontalidad.

Sin reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,87	3,91	4,35	3,37	1,00	3,04
12	0,82	**2,53**	2,99	2,96	1,00	2,11
10	**0,69**	2,60	2,97	-	1,00	**2,09**
8	0,72	2,61	**2,94**	-	1,00	**2,09**
Mejor	0,69	2,53	2,94	-	-	2,05

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,90	3,41	3,55	3,66	1,00	2,62
12	0,88	**2,72**	3,28	3,28	1,00	**2,29**
10	**0,87**	2,91	3,23	-	1,00	2,34
8	**0,87**	2,94	**3,21**	-	1,00	2,34
Mejor	0,87	2,72	3,21	-	-	**2,26**

3.5.3.7.4. Obtención de *outliers*

El último método que se ha implementado para combinar los métodos ASM y LK es de carácter estadístico. Se trata de calcular cada vez una distribución gaussiana con las distancias entre los puntos dados por LK y los puntos dados por ASM y una vez calculada esta distribución, realizar un análisis de los puntos y quitar los *outliers*. Los *outliers* serán los puntos que están muy lejos del centro de la gaussiana calculada.

Para caracterizar los puntos que están fuera de la distribución, se va a tomar un umbral que depende de la desviación estándar de la gaussiana, así, todos los puntos cuya diferencia sea más grande que la media más 1.5 veces la desviación estándar se descartarán.

Sin Reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,81	4,09	4,33	3,10	1,00	3,08
12	0,70	**2,41**	**2,62**	2,59	1,00	**1,91**
10	**0,56**	2,51	2,67	-	1,00	**1,91**
8	0,63	2,49	2,64	-	1,00	1,92
Mejor	0,56	2,41	2,62	-	-	**1,86**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,92	3,46	3,59	3,75	1,00	2,66
12	0,91	**2,76**	3,32	3,32	1,00	**2,33**
10	**0,88**	2,94	3,26	-	1,00	2,36
8	**0,88**	2,97	**3,23**	-	1,00	2,36
Mejor	0,88	2,76	3,23	-	-	**2,29**

3.5.3.8. AAM

Para la obtención de los resultados en AAM la implementación es muy parecida a la de IntraFace y ASM. Los puntos del seguimiento han sido proporcionados por el Trabajo de José Javier Bengoechea[19]. Así, lo único que se ha hecho es variar el programa principal levemente para que el seguimiento consista en coger los puntos proporcionados por José Javier, de tal manera que luego éstos se evalúan igual que en los otros métodos.

El resto de programa, una vez hecho el seguimiento, es el mismo que para Lucas-Kanade.

BU:

PUNTOS	ROLL	YAW	PITCH	Éxito	Promedio
12	4,17	**5,72**	**5,97**	1,00	**5,29**
10	**4,05**	6,11	6,30	1,00	5,49
8	4,06	6,03	6,12	0,99	5,40
Mejor	4,05	5,72	5,97	-	**5,25**

UPNA:

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	1,11	4,19	4,62	4,00	1,00	3,31
12	1,02	2,42	**3,09**	3,68	1,00	**2,18**
10	**0,92**	2,41	3,50	-	1	2,28
8	0,94	**2,38**	3,37	-	1,00	2,23
Mejor	0,92	2,38	3,09	-	-	**2,13**

3.5.3.9. LK + AAM

Conociendo los buenos resultados que se obtienen con el método de LK + ASM se ha visto necesario probar el método con el AAM. La implementación es exactamente igual a excepción de usar los puntos de AAM en vez de ASM, así que no se repetirá en este apartado.

Lo único que cabe mencionar es que por implementarlo en el programa general se prescindirá de hacer el método 3, el de asignación de pesos por cantidad de rotación, ya que los resultados en el apartado anterior son los peores y no aportan nada al estudio.

3.5.3.9.1. Delta inicial-actual

Sin Reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,84	4,79	5,26	2,92	1,00	3,63
12	0,67	2,38	**2,61**	2,36	1,00	1,89
10	**0,46**	**2,13**	2,65	-	1,00	**1,74**
8	0,54	2,17	2,63	-	1,00	1,78
Mejor	0,46	2,13	2,61	-	-	**1,73**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,91	4,04	4,56	3,65	1,00	3,17
12	0,83	**2,32**	**2,99**	3,35	1,00	**2,04**
10	**0,73**	**2,32**	3,34	-	1,00	2,13
8	0,75	**2,32**	3,24	-	1,00	2,10
Mejor	0,73	2,32	2,99	-	-	**2,01**

3.5.3.9.2. Delta inicial-actual previa-actual

Sin Reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,83	4,74	5,22	2,91	1,00	3,60
12	0,66	2,38	2,71	2,39	1,00	1,92
10	**0,46**	**2,13**	2,74	-	1,00	**1,78**
8	0,55	2,18	**2,70**	-	1,00	1,81
Mejor	0,46	2,13	2,70	-	-	**1,76**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,91	4,02	4,54	3,64	1,00	3,16
12	0,82	**2,29**	**3,00**	3,34	1,00	**2,04**
10	**0,73**	**2,29**	3,36	-	1,00	2,13
8	0,75	**2,29**	3,26	-	1,00	2,10
Mejor	0,73	2,29	3,00	-	-	**2,01**

3.5.3.9.3. Obtención de outliers

Sin Reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,78	4,32	5,12	2,95	1,00	3,40
12	0,68	2,41	2,89	2,47	1,00	1,99
10	**0,50**	**2,20**	2,89	-	1,00	**1,86**
8	0,59	2,24	**2,84**	-	1,00	1,89
Mejor	0,50	2,20	2,84	-	-	**1,85**

Reasignado

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
54	0,93	4,08	4,57	3,75	1,00	3,19
12	0,84	**2,36**	**3,07**	3,44	1,00	**2,09**
10	**0,74**	2,38	3,46	-	1,00	2,20
8	0,76	2,38	3,34	-	1,00	2,16
Mejor	0,74	2,36	3,07	-	-	**2,06**

3.5.3.10. LK + ASM + 4 puntos en nariz

En otros proyectos realizados en la UPNA, se ha comprobado que si añadimos cuatro puntos en la nariz, correspondientes al tabique nasal, la medida del POSIT mejora. Para probar si esto se cumple en nuestras implementaciones, vamos a probar dos formas de añadir estos cuatro puntos y utilizar el POSIT.

3.5.3.10.1. Puntos creados con los datos obtenidos

En el caso del IntraFace éste los tiene implementados por defecto, pero en el resto de los casos, cuando se utilizan tanto 54 como 12 puntos, ninguno está en la superficie de la nariz. Para implementar y ver si estos puntos pueden mejorar el POSIT, la primera opción es pasar el vídeo con los puntos tal y como están, pero después del procesado y antes de hallar el POSIT, se calcula mediante triangulación de coordenadas baricéntricas la colocación de los 4 puntos de la nariz.

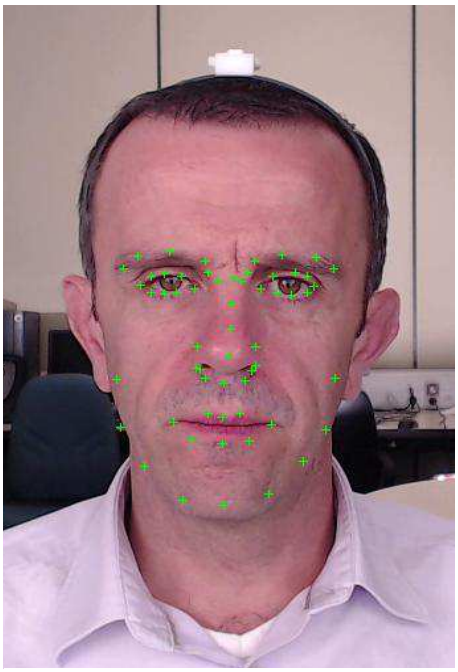


Figura 58: Puntos solución con los 4 puntos del IF añadidos.

Para recordar, en la triangulación se necesitan dos sets de puntos iniciales, en el primer fotograma se calculan las coordenadas baricéntricas de cada punto con respecto a los demás y después se aplican sobre los datos en el resto de los fotogramas. Así, se toman los puntos iniciales que se disponen en el método del IntraFace correspondientes a los 4 puntos de la nariz, y se insertan sobre nuestros fotogramas marcados, de tal manera que se utilizan

58 puntos al final. En la **figura 58** se pueden observar los 58 puntos en el usuario número cinco del primer fotograma.

Después, a partir de los puntos que ya se han obtenido por el método normal recalculamos la situación en la que se encontrarían los 4 puntos de la nariz añadidos en cada fotograma del vídeo. Al igual que se ha explicado en la obtención del nuevo *GroundTruth* del IntraFace, los puntos situados en el tabique nasal no se triangulan correctamente. Esto es porque son puntos que se salen fuertemente del plano imagen y por ello no se cumplen las bases para una buena obtención de coordenadas baricéntricas.

Este cálculo de los nuevos puntos se va a realizar con el algoritmo de LK + ASM con el método 1ns (método 1 no reasignados), por ser el que mejor resultados tiene hasta el momento.

Sin reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
58	0,80	3,41	3,52	3,00	1,00	2,58
16	0,62	**2,34**	2,68	2,31	1,00	1,88
14	**0,53**	2,44	2,63	-	1,00	**1,86**
12n	0,59	2,42	**2,60**	-	1,00	1,87
Mejor	0,53	2,34	2,60	-	-	**1,82**

3.5.3.10.2. Puntos obtenidos del IntraFace

Otra de las comprobaciones que se quieren implementar es, sabiendo que el seguimiento de los puntos de la nariz es bueno (en el caso del IntraFace se considera preciso), ver si esto influye en la obtención de la estimación de la posición y rotación de la cabeza positivamente.

Para ello, se van a utilizar los mejores puntos obtenidos como resultado, el caso de LK + ASM con el método 1 y el mismo pero con AAM, y se añaden los 4 puntos correspondientes a la nariz con el seguimiento del IntraFace. Si estos puntos influyen positivamente, los resultados serán mejores en el cálculo con POSIT.

Sin Reasignar ASM

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
58	0,82	4,15	3,97	3,48	1,00	2,98
16	0,69	2,38	**2,09**	4,13	1,00	1,72
14	**0,54**	**2,15**	2,16	-	0,99	**1,62**
12n	0,60	**2,15**	2,12	-	0.99	1,63
Mejor	0,54	2,15	2,09	-	-	**1,59**

Sin Reasignar AAM

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
58	0,83	4,46	4,49	3,34	1,00	3,26
16	0,69	2,47	**2,11**	4,02	1,00	1,76
14	**0,47**	2,09	2,16	-	1,00	**1,57**
12n	0,55	**2,08**	2,14	-	1,00	1,59
Mejor	0,47	2,08	2,11	-	-	**1,55**

3.5.3.11. LK + IF

En vista de los buenos resultados que proporciona el hecho de añadir los cuatro puntos del tabique nasal, se cree interesante probar a combinar Lucas-Kanade y el IntraFace, con el método de las deltas iniciales y añadiendo los cuatro puntos de la nariz.

Siendo el IntraFace el método que mejor resultados facilita, se quiere observar si se puede mejorar añadiendo Lucas-Kanade con sus puntos iniciales y combinándolo con él. Así, la implementación de este programa consiste en cargar los mismos puntos iniciales que en el IntraFace en el algoritmo de Lucas-Kanade, y realizar el proceso de seguimiento a lo largo de todo el vídeo de tal manera que el cálculo del punto final en cada fotograma es un promedio entre Lucas-Kanade e IntraFace calculado así:

$$\omega_{LK} = 1 - \frac{\delta_{LKinit}}{\delta_{LKinit} + \delta_{IFinit}}$$

$$\text{OutPoint} = \omega_{LK} * (x, y)_{LK} + (1 - \omega_{LK}) * (x, y)_{IF}$$

Como en los resultados anteriores el método funcionaba mejor sin reasignar los puntos de Lucas-Kanade, sólo se han obtenido puntos para el mismo procedimiento.

Sin reasignar

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
49	0,58	2,00	2,79	2,62	1,00	1,79
16	0,88	1,88	2,76	2,71	1,00	1,84
14	0,88	**1,88**	2,76	-	1,00	1,84
12	**0,47**	2,06	**2,22**	-	1,00	**1,58**
Mejor	0,47	1,88	2,22	-	-	**1,52**

4. Análisis de Resultados y Discusión

En la elaboración de resultados hay gran cantidad de parámetros que se utilizan a lo largo de los métodos. Sólo se pondrán los datos que se han variado o que se crean importantes. Si algún dato no aparece, se presupone que se utiliza el proporcionado por defecto.

4.1. BU

Primero se va a comenzar a analizar la base de datos de la Boston University. En general los resultados son peores, quizá por la peor calidad de imagen. Antes de observar los resultados, se explicará brevemente cómo están expuestos.

En la tabla de datos están los errores de POSIT en Roll, Yaw y Pitch de los diferentes grupos de puntos que se han tenido en cuenta. Para esta base de datos son de 49 (grupo 1), de 12 (grupo 2), de 10 (grupo 3) y de 8 (grupo 4). En la **figura 59** se observan las distintas configuraciones de puntos.

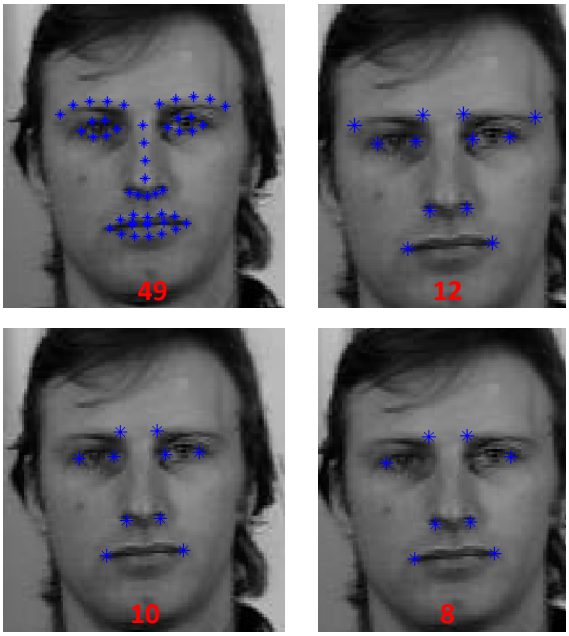


Figura 59: Grupos de puntos en BU.

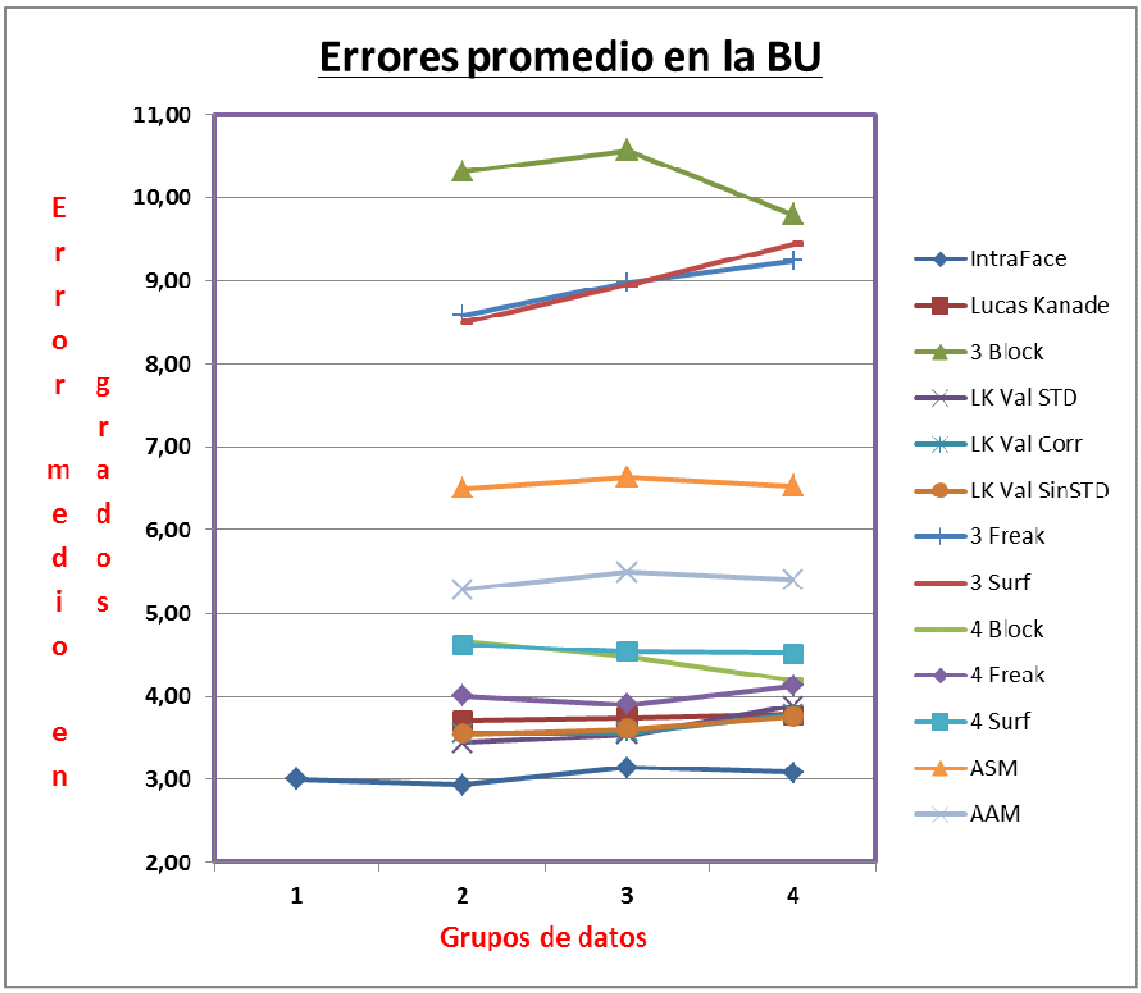
A continuación se presentan los resultados y se hablará sobre ellos.

4.1.1. Tabla de resultados globales de BU

Nº	METHOD	R 49	Y 49	P 49	AV 49	R 12	Y 12	P 12	AV 12	R 10	Y 10	P 10	AV 10	R 8	Y 8	P 8	AV 8
1	**LK**	-	-	-	-	1,87	5,63	3,62	**3,71**	1,86	5,19	4,17	**3,74**	1,95	5,12	4,26	**3,77**
2	**LK VAL NOR.**	-	-	-	-	1,87	4,65	3,78	**3,44**	1,89	4,43	4,27	**3,53**	1,94	4,80	4,90	**3,88**
2	**LK VAL CORR**	-	-	-	-	1,90	5,10	3,66	**3,55**	1,81	4,69	4,18	**3,56**	1,92	4,78	4,66	**3,79**
2	**LK VAL SinSTD**	-	-	-	-	1,87	5,19	3,57	**3,55**	1,86	4,81	4,14	**3,61**	1,94	5,10	4,23	**3,75**
3	**3 FREAK**	-	-	-	-	3,35	9,22	13,20	**8,59**	3,63	9,21	14,11	**8,98**	3,79	9,35	14,58	**9,24**
3	**3 BLOCK**	-	-	-	-	6,56	11,65	12,74	**10,32**	7,03	10,76	13,92	**10,57**	5,49	10,97	12,94	**9,80**
3	**3 SURF**	-	-	-	-	3,95	12,50	9,05	**8,50**	4,24	12,73	9,89	**8,96**	4,06	13,35	10,95	**9,45**
4	**4 FREAK**	-	-	-	-	2,01	5,72	4,28	**4,00**	1,97	5,06	4,67	**3,90**	2,03	5,48	4,86	**4,12**
4	**4 BLOCK**	-	-	-	-	2,37	6,32	5,33	**4,67**	2,10	5,34	5,96	**4,47**	2,04	5,29	5,24	**4,19**
4	**4 SURF**	-	-	-	-	2,52	6,21	5,13	**4,62**	2,66	5,75	5,21	**4,54**	2,28	6,15	5,11	**4,51**
6	**IntraFace**	1,73	3,80	3,47	**3,00**	1,77	3,64	3,40	**2,94**	1,79	3,78	3,86	**3,14**	1,79	3,71	3,73	**3,08**
7	**ASM**	-	-	-	-	3,86	8,42	7,24	**6,51**	3,86	8,93	7,11	**6,64**	3,84	8,95	6,80	**6,53**
9	**AAM**	-	-	-	-	4,17	5,72	5,97	**5,29**	4,05	6,11	6,30	**5,49**	4,06	6,03	6,12	**5,40**

4.1.2. Gráfica de BU

Una gráfica puede dar una idea más acertada de cómo están situados los datos entre sí. Se presenta una con todos los métodos evaluados en la BU:



4.1.3. Discusión de resultados de BU

Después de los datos que hemos obtenido, se pueden realizar varias observaciones respecto a ellos.

La primera y bastante importante, es que en ninguno de los casos se consigue un error promedio menor a **2,94º.**

La segunda idea que se puede extraer es que el IntraFace es el mejor algoritmo para la BU, obteniendo los mejores resultados y en tiempo real, lo cual lo convierte en un sistema muy potente y a tener en cuenta.

Con respecto a los algoritmos de AAM y ASM, se deduce que para la BU sus resultados son pésimos, siendo en el primer caso en torno a 5,5º y en el segundo en torno a 6,5º. En cambio y como veremos más adelante, para la UPNA los resultados son mejores. Esto es porque los procedimientos de AAM y ASM han sido entrenados con imágenes de la base de datos de la UPNA, y esto es un claro condicionante de estos resultados.

En cuanto al algoritmo de Lucas-Kanade y sus mejoras con respecto a la validación de puntos, todos los métodos son muy parecidos. Es cierto que con la validación STD se obtienen los mejores resultados, pero la diferencia es bastante pequeña. Los errores de estos procedimientos se mueven en torno a 3,44º.

Los métodos de corrección con descriptores para esta base de datos se descartan totalmente, tienen los errores más altos y su coste computacional es elevado. En el caso de comparar los descriptores del fotograma actual con el inicial los resultados son algo mejores, pero aun así insuficientes para lo que se requiere como mínimo. Los mejores errores están en torno a 4,1º.

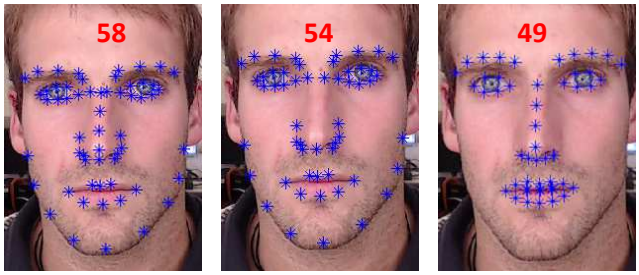
4.2. UPNA

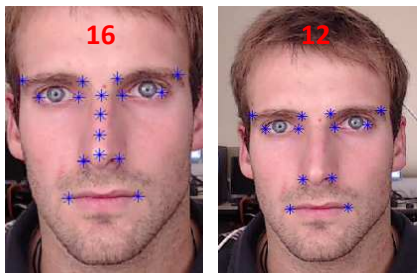
Por último se va a comentar la base de datos de la Universidad Pública de Navarra. En general los resultados son mejores.

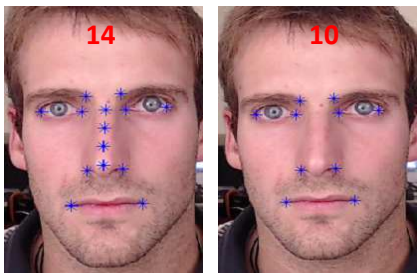
En la tabla de datos están los errores de POSIT en Roll, Yaw y Pitch de los diferentes grupos de puntos que se han tenido en cuenta. Para esta base de datos varían según el método, siendo 58,54 ó 49 (grupo 1), de 16 ó 12 (grupo 2), de 14 ó 10 (grupo3) y de 12n (4 puntos de la nariz) u 8 (grupo 4). Esto es porque en un principio no se tenían pensado colocar los cuatro puntos correspondientes al tabique nasal y, al ver que los resultados mejoran notablemente, se decidió tenerlo en cuenta al final.

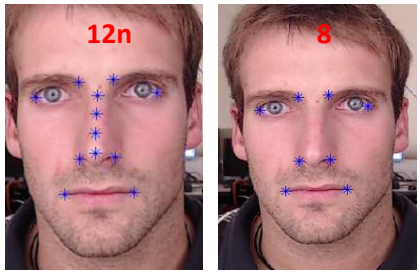
En la **figura 60** se observan las distintas configuraciones de puntos.

Relación de puntos por grupo y método









Método	G 1	G 2	G 3	G 4
LK	54	12	10	8
LK VAL STD	54	12	10	8
LK VAL CORR	54	12	10	8
LK VAL sinSTD	54	12	10	8
3 FREAK	54	12	10	8
3 BLOCK	54	12	10	8
3 SURF	54	12	10	8
4 FREAK	54	12	10	8
4 BLOCK	54	12	10	8
4 SURF	54	12	10	8
INTRAFACE	49	12	10	8
ASM	54	12	10	8
LK-ASM init ns	54	12	10	8
AAM	54	12	10	8
LK - AAM init ns	54	12	10	8
LK-ASM init ns - IF	54	12	10	8
LK - IF	49	16	14	12n

Figura 60: Grupos de puntos en la UPNA.

4.2.1. Tabla de resultados globales UPNA

Seguimiento:

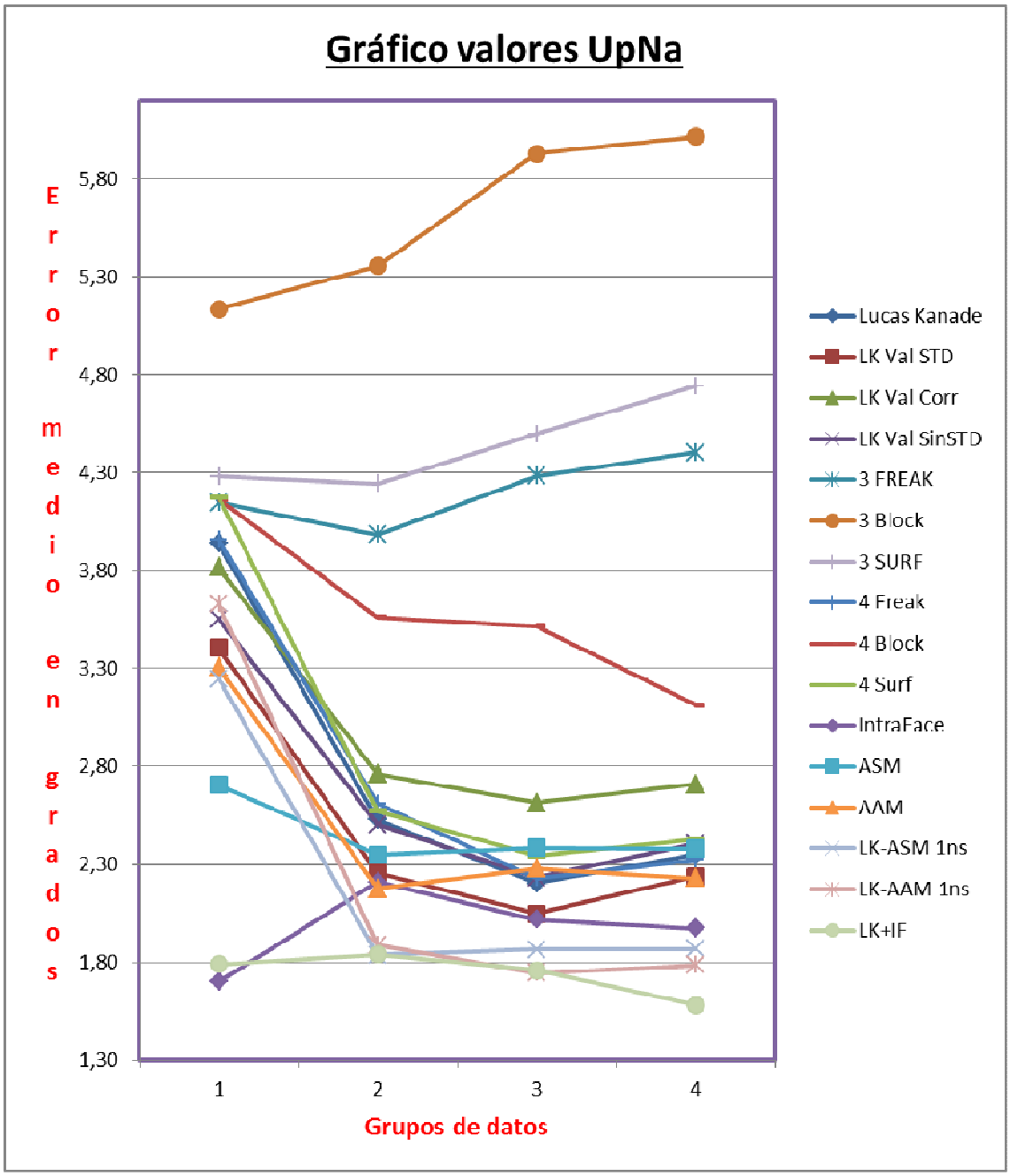
El error de seguimiento se calcula únicamente en la base de datos de la UPNA, porque sólo en ésta se poseen los datos de dónde están situados los puntos realmente (el *GroundTruth* de seguimiento). Los resultados son:

Método	E 54	E 49	E 12
LK	4,20	-	3,28
LK Val STD	3,70	-	2,85
LK Val Corr	4,02	-	3,17
LK Val sinSTD	3,92	-	3,13
3 FREAK	8,03	-	6,60
3 BLOCK	10,80	-	10,61
3 SURF	8,37	-	7,49
4 FREAK	4,75	-	3,17
4 BLOCK	5,33	-	4,50
4 SURF	4,80	-	3,02
IntraFace	-	2,70	2,45
ASM	3,90	-	3,50
LK-ASM 1ns	3,07	-	2,51
AAM	4,00	-	3,68
LK-AAM 1ns	2,92	-	2,36
LK-ASM 1ns – IF	3,48	-	4,13
LK-IF	-	2,62	2,71

POSIT

	METHOD	R 1	Y 1	P 1	AV 1	R 2	Y 2	P 2	AV 2	R 3	Y 3	P 3	AV 3	R 4	Y 4	P 4	AV 4
1	LK	1,30	5,06	5,46	**3,94**	1,33	2,92	3,34	**2,53**	0,77	2,83	3,02	**2,20**	0,95	2,95	3,14	**2,35**
2	LK VAL STD	1,16	4,57	4,49	**3,41**	1,09	2,57	3,11	**2,26**	0,70	2,60	2,85	**2,05**	0,84	2,78	3,10	**2,24**
2	LK VAL CORR	1,28	4,76	5,41	**3,82**	1,59	2,91	3,79	**2,76**	1,27	2,97	3,61	**2,62**	1,40	3,04	3,69	**2,71**
2	LK VAL sinSTD	1,24	4,95	4,47	**3,55**	1,28	2,95	3,29	**2,51**	0,78	2,88	3,03	**2,23**	0,97	3,07	3,18	**2,41**
3	3 FREAK	1,74	4,76	5,94	**4,15**	2,16	4,30	5,49	**3,98**	2,36	4,87	5,62	**4,28**	2,60	4,94	5,67	**4,40**
3	3 BLOCK	2,46	5,13	7,81	**5,14**	3,24	4,97	7,86	**5,35**	3,07	6,11	8,60	**5,93**	3,18	6,32	8,55	**6,02**
3	3 SURF	1,91	5,14	5,79	**4,28**	2,47	5,09	5,18	**4,25**	2,52	5,94	5,03	**4,50**	2,62	6,24	5,37	**4,74**
4	4 FREAK	1,47	5,07	5,35	**3,96**	1,49	2,93	3,41	**2,61**	0,89	2,54	3,26	**2,23**	1,01	2,63	3,34	**2,32**
4	4 BLOCK	1,68	4,57	6,24	**4,16**	1,81	4,00	4,88	**3,56**	1,33	3,84	5,38	**3,52**	1,17	3,56	4,61	**3,11**
4	4 SURF	1,45	5,22	5,85	**4,17**	1,31	3,13	3,28	**2,57**	0,84	2,91	3,29	**2,34**	0,88	2,97	3,44	**2,43**
6	INTRAFACE	0,50	2,08	2,52	**1,70**	0,64	1,89	4,10	**2,21**	0,42	2,10	3,53	**2,02**	0,44	1,96	3,51	**1,97**
7	ASM	0,92	3,53	3,68	**2,71**	0,91	2,78	3,36	**2,35**	0,89	2,97	3,30	**2,39**	0,89	2,97	3,28	**2,38**
8	LK-ASM 1ns	0,82	4,34	4,58	**3,25**	0,66	2,27	2,59	**1,84**	0,53	2,44	2,63	**1,86**	0,59	2,42	2,60	**1,87**
9	AAM	1,11	4,19	4,62	**3,31**	1,02	2,42	3,09	**2,18**	0,92	2,41	3,50	**2,28**	0,94	2,38	3,37	**2,23**
10	LK - AAM 1ns	0,84	4,79	5,26	**3,63**	0,67	2,38	2,61	**1,89**	0,46	2,13	2,65	**1,74**	0,54	2,17	2,63	**1,78**
11	LK-ASM 1ns - IF	0,82	4,15	3,97	**2,98**	0,69	2,38	2,09	**1,72**	0,54	2,15	2,16	**1,62**	0,60	2,15	2,12	**1,63**
12	LK - IF	0,58	2,00	2,79	**1,79**	0,88	1,88	2,76	**1,84**	2,06	2,22	1,00	**1,76**	0,47	2,06	2,22	**1,58**

4.2.2. Graficas de UPNA



4.2.3. Discusión de resultados UPNA

Se van a comentar los resultados obtenidos para la base de vídeos de la UPNA. Lo primero que se observa es que el rango de valores en el que nos movemos es muy

diferente, en la BU los datos se mueven entre 2,94º y 10,5º; en la UPNA los valores van desde 1,57º hasta 6º. Esto indica que los resultados en esta base de datos son mejores.

En cuanto al mejor algoritmo, parece que los mejores resultados se dan para el de Lucas-Kanade combinado con IntraFace, siendo el mejor valor para el grupo de **12n** puntos utilizando el tabique nasal. **Este error es de 1,58º de promedio**.

De nuevo, la utilización de los descriptores no aporta resultados positivos, siendo para el método 3, el que crea nuevos descriptores cada fotograma, el grupo con peores resultados. En el caso del método 4, los resultados se acercan a los demás pero tampoco mejoran Lucas-Kanade ya que están en torno a 2,23º. Esto y el hecho de que necesiten un gran tiempo computacional hacen de ellos métodos poco adecuados.

El ASM y el AAM dan resultados por sí solos bastante aceptables, 2,35º el mejor promedio para ASM y 2,18º para AAM. En cuanto a estos métodos, al ser algoritmos que mantienen la apariencia y la coherencia espacial entre los puntos, el conjunto de ellos se asemeja fuertemente a una cara y los resultados son más robustos. Sin embargo, y como se ha explicado anteriormente, al aumentar significativamente el error cuando la máscara se desvía de la cara, no mejoran el resultado que proporciona Lucas-Kanade.

En el momento en el que se añade la combinación con LK y la corrección de puntos (de tal manera que nunca se desvíen de la cara), el ASM y el AAM reducen sus errores hasta 1,86º y 1,74º respectivamente, siendo unos resultados muy aceptables. El problema que se puede hallar en ellos es su tiempo computacional, bastante elevado.

Por último, el IntraFace ofrece el segundo mejor resultado, teniendo un promedio de 1,70º.

En cuanto a los errores de seguimiento, en todos los casos los mejores resultados de la estimación de la posición y rotación de la cabeza tienen los menores errores de seguimiento. También se puede observar que los doce puntos característicos tienen de media errores más bajos y se siguen mejor en todos los algoritmos.

4.3 Ideas generales

Hay una serie de puntos en cuanto a resultados generales que se deberían comentar para futuros trabajos en esta línea.

Uno es el hecho de que el IntraFace es el método que en general da mayor prestación en cuanto a calidad y tiempo de procesado. Funciona en tiempo real. El único problema que presenta es la inestabilidad de los resultados.

Otro es que combinar IntraFace con Lucas-Kanade también da muy buenos resultados, ejecutándose a tiempo real con un buen ordenador y reduciendo el error del IntraFace en un 7%.

También cabe añadir que en cuanto a la cantidad de puntos, en general una mayor cantidad de puntos no da mejores resultados, a menos que la totalidad de ellos tenga un error de seguimiento muy bajo (IntraFace o LK+IF). Por ello, en general es mejor utilizar los 12 puntos planteados inicialmente o los 12n, esto es, los que incluyen cuatro puntos del tabique nasal. Esto es porque los otros puntos (sobre todo aquellos que están en la mandíbula) tienen un error de seguimiento muy alto y estropean los resultados.

En cuanto a la inclusión de los puntos del tabique nasal, parece que mejora en gran medida los resultados, de tal manera que el mejor resultado se da cuando están incluidos. Esto puede ser porque al salirse fuertemente de la coplanaridad del resto de los puntos de la cara, ayuda en gran medida al cálculo del POSIT, principalmente en las rotaciones fuera del plano imagen (Pitch y Yaw). El problema que tienen dichos puntos es que no son muy característicos en la imagen, no tienen ningún color ni forma especial, ni hacen esquina con otras partes de la cara, por lo que podrían resultar, a priori, complicados de seguir con Lucas-Kanade u otro tipo de algoritmos de seguimiento.

Por comprobación y para ver si esta suposición era cierta, se calculó unos resultados añadiendo estos cuatro puntos al método original de Lucas-Kanade. Los resultados son sorprendentes:

PUNTOS	ROLL	YAW	PITCH	Error Seg.	Éxito	Promedio
58	1,29	4,50	4,68	-	1,00	3,49
16	1,33	**2,20**	2,71	-	1,00	2,08
14	**0,88**	2,34	2,32	-	1,00	**1,85**
12n	1,08	2,35	**2,31**	-	1,00	1,91
Mejor	0,88	2,20	2,31	-	-	**1,80**

Esto da un nuevo enfoque en cuanto a puntos a seguir. Con Lucas-Kanade en este caso el error de seguimiento no se puede tener en cuenta porque el nuevo *GroundTruth* que se calcula para los puntos de la nariz la triangulación no es válida, pero en cuanto al POSIT se llega a un valor más que aceptable, superando a casi todos los métodos.

5. Conclusiones

En un principio, el proyecto constaba de dos partes bien diferenciadas. La primera parte, la realización de un escaneo digital en tres dimensiones de un modelo real, que consistía en un busto o maniquí. Esta parte se basaba en el escaneo con dos máquinas distintas, la Konica Minolta, más potente pero más cara, y DAVID LaserScanner, más barato y más complicado. Este proyecto también contiene una pequeña comparación de entre ambos sistemas de escaneo.

Una vez hecho el escaneo digital, también se realiza un registro de los 54 puntos característicos usados en otros proyectos para saber cuáles pertenecen al modelo. Con esto se logra una serie de datos de los puntos tridimensionales del modelo y los puntos característicos del mismo, tanto los marcados con el sensor trakSTAR cómo sus correspondientes más cercanos al modelo.

En la segunda parte del proyecto, algo más amplia, se cumplen una serie de objetivos. Primero se participa en la grabación de la Base de vídeos de la UPNA como un usuario más y también en algún calibrado y en los vídeos de Dasha.

Después llega la evaluación de algoritmos. Se llegan a implementar 11 tipos distintos de ellos, con algunos extra (LK con 58 puntos) y se realiza un análisis exhaustivo de ellos, calculando varios errores de seguimiento (distancias euclídeas, robustez de cada punto, error por fotograma de cada vídeo) y varios errores de POSIT (error en Roll, Yaw, Pitch, error promedio y éxito en cada vídeo).

Con esta evaluación se obtienen varias conclusiones de gran interés. La primera es que tener los cuatro puntos del tabique nasal entre los característicos mejora los resultados de manera cuantitativa ya que aporta una gran mejora en el análisis del Yaw y el Pitch de la cabeza. La segunda es que el mejor sistema en cuanto a precisión es el formado por la combinación de Lucas-Kanade con IntraFace, siendo el error mínimo de **1,58º**.

También se observa que, dependiendo del algoritmo, hay mejores errores de Roll, Yaw o Pitch dependiendo del grupo de puntos que se tomen. Así, si se toman para LK-IF los mejores valores se reduce el error promedio a 1,52º.

Por último, resulta claro al ver las diferencias de los rangos entre los que se mueven los errores que en una base de datos con más calidad de imagen, como es el caso de la UPNA, se reducen en gran manera los errores de POSIT.

6. Bibliografía.

La bibliografía utilizada para la realización de este PCF se detalla a continuación:

- Libros:

1. MULTIPLE VIEW GEOMETRY IN COMPUTER VISION. Richard Hartley and Andrew Zisserman. Cambridge University Press, March 2004

- Otros artículos:

[1] Documentación oficial de MATLAB.

[2] Murphy-Ghutorian, Erik, Manubhai Trivedi, Mohan, IEEE. "Head Pose Estimation in Computer Vision: a Survey", April 2009. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.159.8306&rep=rep1&type=pdf

[3] Artículo original sobre el algoritmo de Lucas-Kanade:
D. Lucas, Bruce, Kanade, Takeo. "An Iterative Image Registration Technique with an Application to Stereo Vision", 1981.
http://cseweb.ucsd.edu/classes/sp02/cse252/lucaskanade81.pdf

[4] Artículo sobre el algoritmo de Lucas-Kanade:
Bouguet, Jean-Yves. "Pyramidal Implementation of the Lucas-Kanade Feature Tracker Description of the algorithm", 2000.
http://robots.stanford.edu/cs223b04/algo_tracking.pdf

[5] Power-Point sobre optical flow:
Hoiem, Derek. "Feature Tracking and Optical Flow", Power Point of the University of Illinois,2012.
http://courses.engr.illinois.edu/cs543/sp2012/lectures/Lecture%2008%20-%20Feature%20Tracking%20and%20Optical%20Flow%20-%20Vision_Spring2012.pdf

[6] Power-Point sobre optical flow:
Marcon, Marco. “Structure from Motion Optical Flow, Feature Tracking, Normal Flow”. Power point of Politecnico di Milano, 2003.
http://www-dsp.elet.polimi.it/share1/Marcon/Courses/PhD/3DSFVM/firstLesson_Optical%20Flow.pdf

[7] Artículo sobre el POSIT:
DeMenthon, Daniel F., S.Davis, Larry, “Model-Based Object Pose in 25 Lines of Code”. University of Maryland, 1995.
http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=347E01840E649BC0FCB8E22DFB228873?doi=10.1.1.65.306&rep=rep1&type=pdf

[8] Aplicaciones del seguimiento de objetos:
http://es.wikipedia.org/wiki/Seguimiento_de_objetos#Aplicaciones
http://www.fxguide.com/fxpodcasts/fxpodcast-dr-mark-sagar/
http://es.wikipedia.org/wiki/Tomograf%C3%ADa_por_emisi%C3%B3n_de_POSITrones

[9] Rotation matrix
G. Slabaugh, Gregory, “Computing Euler angles from a rotation matrix”.
https://truesculpt.googlecode.com/hg-history/38000e9dfece971460473d5788c235fbbe82f31b/Doc/rotation_matrix_to_euler.pdf

[10] Registro:
http://en.wikipedia.org/wiki/Iterative_closest_point

Iterative Closest Point program by Wilm, Jakob.
http://www.mathworks.es/matlabcentral/fileexchange/27804-iterative-closest-point

Finite Iterative Closest Point program, by Kroon, Dirk-Jan.
http://www.mathworks.es/matlabcentral/fileexchange/24301-finite-iterative-closest-point

[11] SURF
http://en.wikipedia.org/wiki/SURF

Bay, Herbert, Tuytelaars, Tinne, Gool, Luc Van, “SURF: Speeced Up Robust Features”. ETH Zurich 2008.
http://www.vision.ee.ethz.ch/~surf/eccv06.pdf

[12] FREAK
Alahi, Alexandre, Ortiz, Raphael, Vandergheynst, Pierre, “FREAK: Fast Retina Keypoint”. Ecole Polytechnique Fédérale de Lausanne, Switzerland, 2012.
http://infoscience.epfl.ch/record/175537/files/2069.pdf

[13] SIFT
G. Lowe, David, “Distinctive Image Features from Scale-Invariant Keypoints”. University of British Columbia, Canada, 2004.
http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[14] Programa SIFT
Feature points in image, Keypoint extraction program by Kovnatsky, Artiom, 2014.
http://www.mathworks.com/matlabcentral/fileexchange/29004-feature-points-in-image-keypoint-extraction

[15] INTRAFACE
IntraFace project, Human Sensing Laboratory – Robotics Institute – CMU and university of Pittsburgh.
http://www.humansensing.cs.cmu.edu/IntraFace/about.html

[16] ASM
http://en.wikipedia.org/wiki/Active_shape_model

[17] AAM
http://en.wikipedia.org/wiki/Active_appearance_model

[18] Modelo BFM
Paysan, Pascal, Knothe, Reinhard, Amberg, Brian, Romdhani, Sami, Vetter, Thomas, “A 3D Face Model for Pose and Illumination Invariant Face Recognition”. Genova, Italy, 2009.
http://faces.cs.unibas.ch/bfm/main.php?nav=1-2&id=downloads

[19] Bengoechea Irañeta, José Javier, “Comparativa de Algoritmos de visión monocular para la estimación de la posición de la cabeza”, Universidad Pública de Navarra, 2014.

[20] Echeverría, Rebeca. “Desarrollo de una base de datos de posiciones 3D de la cabeza empleando el sensor TrakSTAR 3D GUIDANCE STUDIO”. Universidad Pública de Navarra.

[21] Prasad Pawan, B. H., Aravin, R. “A Robust Head Pose Estimation System for Uncalibrated Monocular Videos”. IIT Madras, 2010.

Fotos:

- http://www.geekmomprojects.com/wii-nunchuck-controlled-servo-motors/

- https://www.ri.cmu.edu/research_project_detail.html?project_id=629&menu_id=261